SYMBOLIC NETWORK ANALYSIS PROGRAM FOR LARGE SCALE SENSITIVITY ANALYSIS

by

RASHMI GUPTA

DEPARTMENT OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY KANPUR

JULY 1977

SYMBOLIC NETWORK ANALYSIS PROGRAM FOR LARGE SCALE SENSITIVITY ANALYSIS

A Thesis Submitted
In Partial Fulfilment of the Requirements
for the Degree of
MASTER OF TECHNOLOGY

by

RASHMI GUPTA

to the

DEPARTMENT OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY KANPUR

JULY 1977

EE-1977-M-GUP-SYM

CERTIFICATE

Certified that this work 'Symbolic Network Analysis Program for Large Scale Sensitivity Analysis', by Rashmi Gupta has been carried out under my supervision and this work has not been submitted elsewhere for a degree.

T. Lakshmi Viswanathan

July, 1977

Dr. T. L. Viswanathan
Assistant Professor
Department of Electrical Engineering
Indian Institute of Technology
Kanpur

ACKNOWLEDGEMENTS

I wish to express my deep sense of gratitute to Dr.T.L.Viswanathan for her inspiring guidance and cherished encouragement in the work at all stages of the progress.

I thank Dr.Narsingh Deo for several useful discussions. I thank Head of the Computer Centre and the staff for providing facilities on the IBM 7044 and IBM 1800 systems.

I would like to express my deep sense of gratitude to Mr.R.Pandey (Typist), Mr. B.N.Srivastava (Draughtsman) and Mr. Triveni Tiwari (Gestetner Operator).

Rashmi Gupta

LIST OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

The symbolic network analysis program generates network functions such as voltage gain, current gain, transconductance and transresistance etc. of a two port network. These network functions are the ratios of two polynomials which are functions of the complex frequency S containing different symbols such as R, L or C etc. as the coefficients. This program was initially developed by Dr. P.M. Lin and G.E. Alderson. It has been modified to be able to run it in IBM 7044 and IBM 1800. Additional facilities of multiinput, multioutput and frequency response plotting have been incorporated. The frequency response plotting facility has been used to demonstrate the utility of the program for large scale sensitivity analysis.

I. INTRODUCTION

I.1 Introduction

The great majority of computer aided linear circuit analysis programs belong to the class of numerical programs i.e. output is some numerical value. Such is the case with well known programs like ECAP II (Electronic circuit analysis program), SCAP, ANP3, BELAC etc. But a few programs like ANP1[1], NASAP[2] and CORNAP[3] have been developed which can generate network functions as rational functions of 'S'. The program 'SNAP' (Symbolic network-analysis program)[4] was developed by Dr. P.M.Lin and G.E.Alderson for generating symbolic network functions. By a symbolic network function we mean rational functions like

$\frac{V_{out}}{V_{in}}$, $\frac{V_{out}}{I_{in}}$, $\frac{I_{out}}{I_{in}}$ and $\frac{I_{out}}{V_{in}}$ where V_{out} and I_{out} are the

output variables, V_{in} and I_{in} are input variables associated with a two port network. These are ratios of the two polynomials in 'S' containing different symbols as coefficients. For example,

$$\frac{V_{out}}{V_{in}} = \frac{1+2R_1\ CS + C^2R_1R_2S^2}{1+C(R_1+2R_2)S+C^2R_1R_2S^2} \qquad (I.1)$$

I.2 Importance of Symbolic Network Analysis Program (SNAP)

The importance of symbolic analysis is due to the following points

1. Insight
2. Improvement of accuracy of claculation
3. Sensitivity analysis
4. Large scale parameter variation analysis
5. Iterative piecewise linear analysis of resistive networks

Insight

For a small network with all elements in the symbolic form or for a large network with only a few of the network elements represented in symbolic form, the network function would be a relatively simple one. Such an expression can provide better insight than numerical solutions, Consider a simple case of a common collector transistor stage. The voltage gain of such a transistor stage under the assumption that r_c tends to infinity can be shown to be

$$A_V = \frac{V_o}{V_{in}} = \frac{R_L}{(1-\alpha)\ r_b+r_e + R_L} \tag{I.2}$$

where α, r_e, r_b and r_c are the T parameters of the transistor and R_L is the load resistance. By inspection of the above symbolic network function it is clear that voltage gain (A_V) is positive and less than one and very close to one, provided that R_L is much greater than $[(1-\alpha)r_b+r_e]$. Without a symbolic network function the above conclusion can only be reached after the analysis of many numerical cases and even after that some degree of uncertainty exists.

Improvement of Accuracy of Calculations

In the analysis of electrical networks using digital computers, there are several important sources of numerical errors. Among these are the round off error and the loss of significance error. The former is due to finite word length of the machine and the latter occurs during floating point addition of two numbers of opposite sign but comparable magnitude . By the proper use of symbolic parameters, the accuracy of the final result of the calculations can be greatly improved.

To demonstrate how a symbolic program can be used to effectively control round off error, consider the differential amplifier shown in figure I.1 and I.2. In figure I.2 the branches and nodes are numbered.

The network function $\frac{I_{out}}{V_{in}}$ is given by

$$\frac{I_{out}}{V_{in}} = \frac{\frac{R_1R_3A_2}{(R_E)^2R_2} - \frac{R_1R_3A_1}{(R_E)^2R_2} + \frac{R_1R_3A_1}{(R_E)^2R_2} - \frac{R_1A_1}{R_E\ R_2} - \frac{R_3R_1A_2}{(R_E)^2R_2} + \frac{R_1A_2}{R_E\ R_2}}{\frac{R_1}{R_2} + \frac{R_1}{R_2} + \frac{R_3}{R_E} + \frac{R_1R_3}{R_2R_E} + \frac{R_1R_3}{R_2R_E} + \frac{R_1R_3}{R_2R_E} + \frac{R_1R_3}{R_2R_E}}$$

Let $A_2 = A_1$, $R_1 = 5K$, $R_2 = 15K$, $R_3 = 10K$, $R_E = 25$ ohms

Evaluating the numerator and the denominator by summing the terms in the order given in the above sets keeping each number generated to 8 significant digits, we get

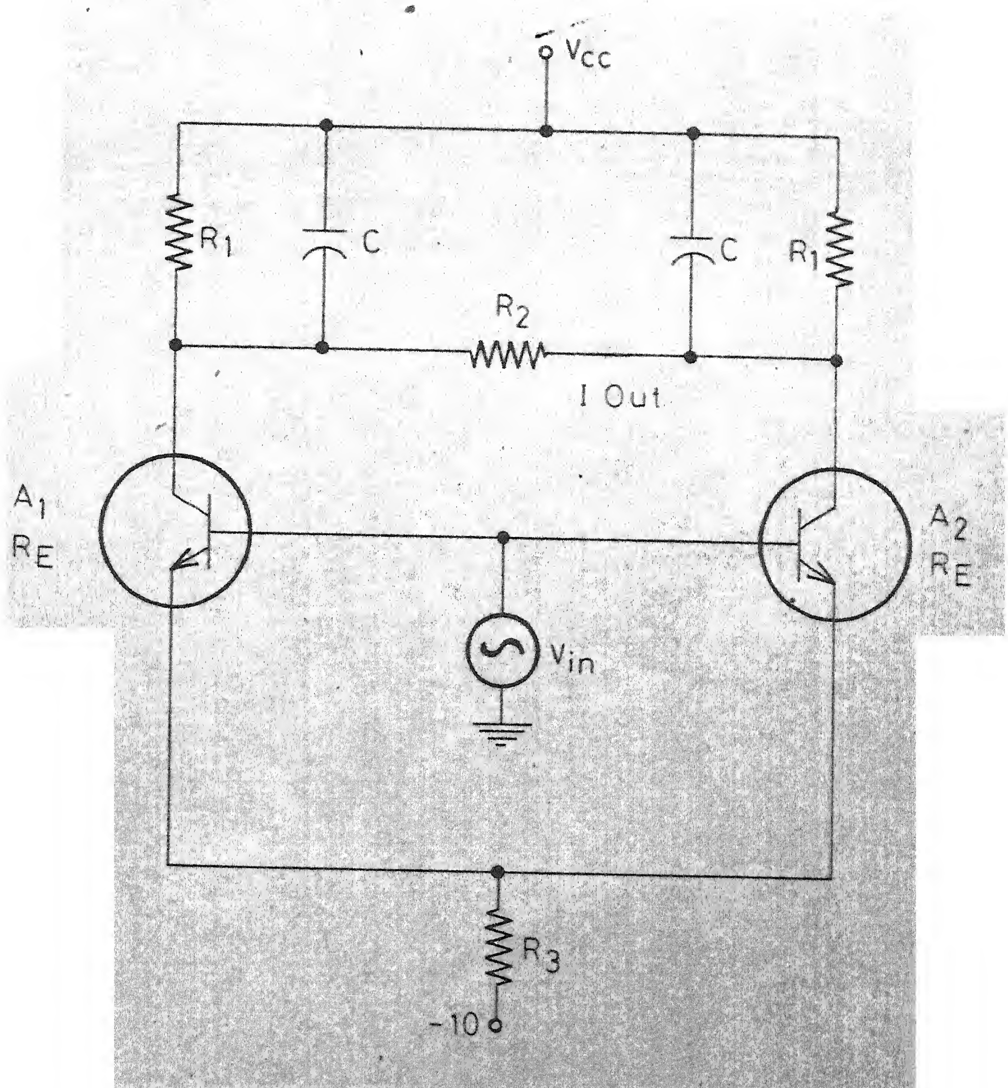

Figure I.1 Differential amplifier.

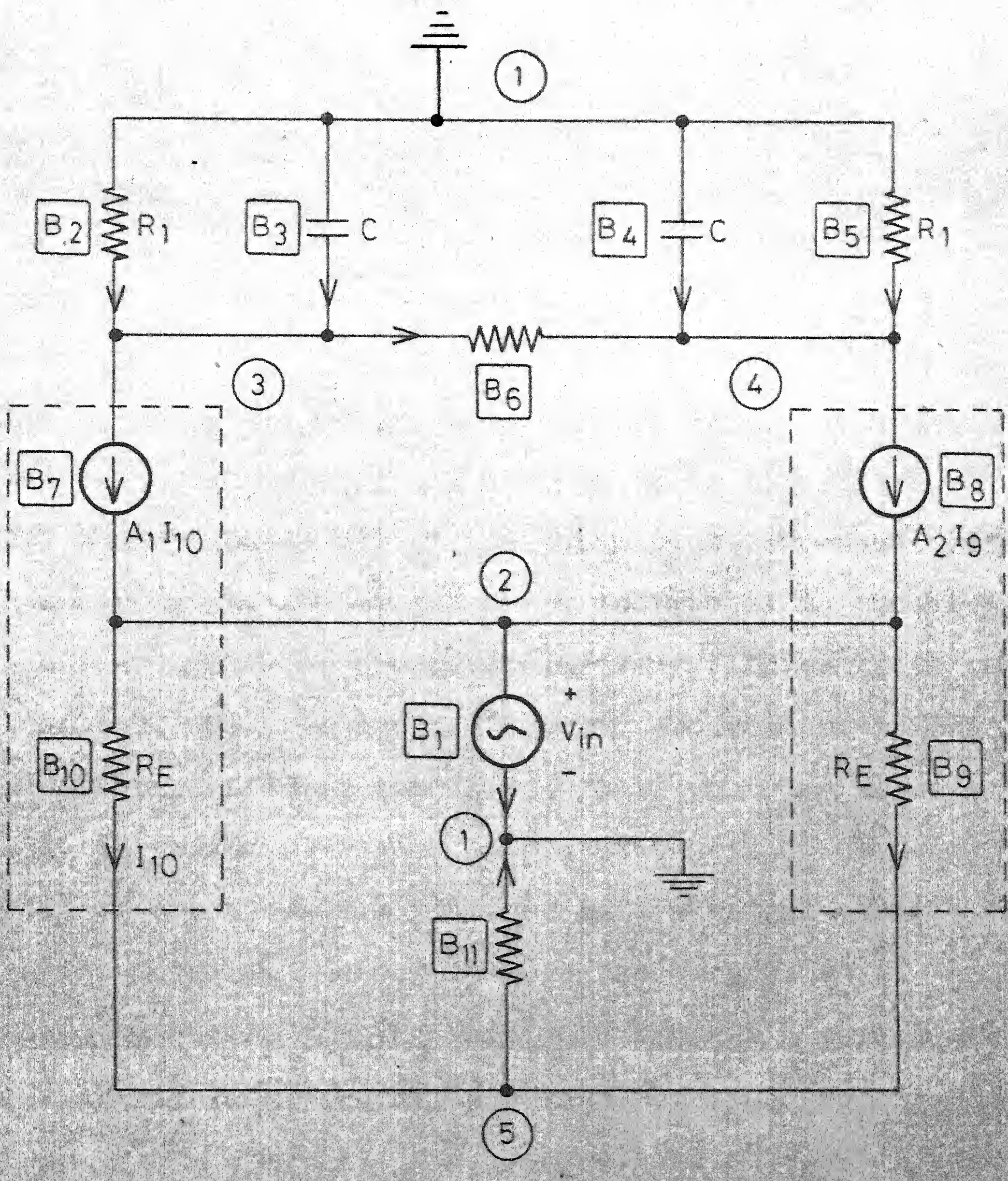

Figure 1.2 Equivalent circuit of differential amplifier.

$$\text{Numerator} = A_1 [\, 5.3333333 - 5.3333333 + 5.3333333 - 0.13333333 - 5.3333333 + 0.13333333 \,]$$

$$= 3.3\text{x}\ 10^{-8}\ A_1$$

and

$$\text{Demominator} = 1335$$

Thus $\left.\frac{I_{out}}{V_{in}}\right|_{S=0} = \frac{3.3\text{x}10^{-8}}{1335}\ A_1$

which is incorrect, since numerator is zero. Although the above transfer function was derived using Signal Flow Graph (SFG) theory, round off errors which cause erroneous results can occur in any computer program restricted to numerical evaluation and these are generally very difficult to predict or control. Because round off error enhancement in the evaluation of network functions often occurs as a result of widely seperated values of some of the network elements, one method of error control would be to leave such element values in symbolic form. This technique can be applied to the above example by noting that R_E should be kept as a symbol since its value is considerably less than the other resistance values. Thus keeping R_E as a symbol and revaluating the numerator gives

$$A_1 \left[\frac{3333.3333}{(R_E)^2} - \frac{3333.3333}{(R_E)^2} + \frac{3333.3333}{(R_E)^2} - \frac{.33333333}{R_E} - \frac{3333.3333}{(R_E)^2} + \frac{.33333333}{R_E} \right]$$

That is $\left.\frac{I_{out}}{V_{in}}\right|_{S=0} = 0$

(NOTE: For details of round-off error see Appendix A)

Sensitivity Analysis

The sensitivity of system performance with respect to changes in component characteristic is a very important consideration in the design of systems. Sensitivity analysis is carried out in numerical programs by following any one of the wellknown methods, like the adjoint network method. But use of symbolic network functions for sensitivity analysis gives a good insight to isolate the important parameters to which the network response is more sensitive. Symbolic network functions also give exact solutions.

For example, the voltage gain of a common emitter transistor stage has been given by equation I.2. The sensitivity of A_v with respect to α is obtained by knowing

$$\frac{\delta A_v}{\delta\alpha} = \frac{R_L\, r_b}{[\ (1-\alpha)r_b+r_c+R_L]^2} \qquad (I.3)$$

This expression gives the exact solution for $\frac{\delta A_v}{\delta\alpha}$ in terms of all the parameters like α, R_L etc.

Similarly sensitivity with respect to other parameters can be calculated. Higher order sensitivities such as $\frac{\delta^2 A_v}{\delta\alpha.\delta R_L}$, may be obtained by repeated differentiation.

Large Scale Parameter Variation Analysis

The sensitivity function of the type given by equation I.3 is applicable only when changes in system parameters are of incremental nature. When relatively large changes occur in a parameter, that parameter is put in the symbol form and

network function is found by putting different values of that parameter.

For example, if the voltage gain A_v of common emitter transistor amplifier is to be calculated for different values of R_L, let R_L take successive values such as 1K, 2K------10K. Then ten analysis of the complete network would be necessary for a numerical program. But if gain function is derived with R_L kept as a symbol, as given in equation I.2, it is only necessary to evaluate the gain function ten times which is a much simpler task.

Iterative Piecewise Linear Analysis of Resistive Nonlinear Networks

A part of this powerful analysis technique requires the solution of a resistive linear network where some resistances and some d.c. sources are kept in symbol form.

For example consider the network shown in figure I.3 with nonlinear resistors R_1 and R_2 characterized respectively by the i-v curves shown in figure I.4 and I.5. We want to find out currents in resistors R_1 and R_2.

By applying the iterative piecewise linear method, we replaced the two nonlinear resistors by their iterative Thevenin equivalent circuits as shown in figure I.6. The nonlinear resistors take different values of voltage (E) and resistance (R) in three different segments as given by table I.1.

By symbolic network analysis program, we get currents in resistors R_1 and R_2 as I_1 and I_2 respectively.

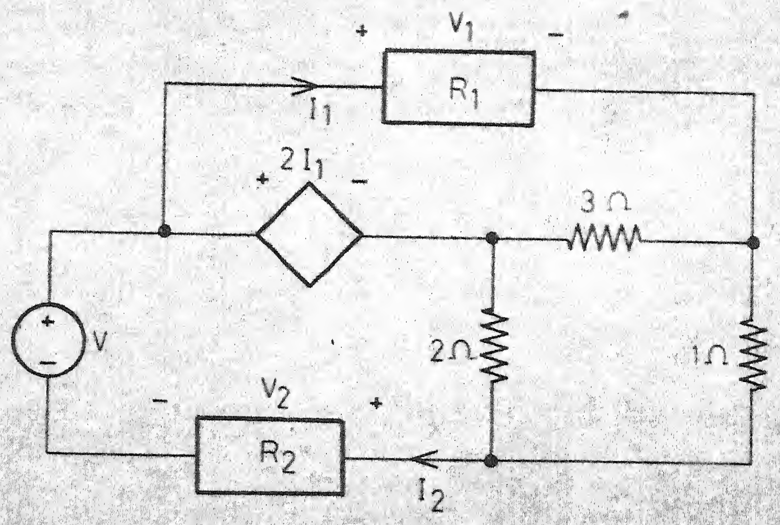

Figure I.3 Nonlinear resistive circuit.

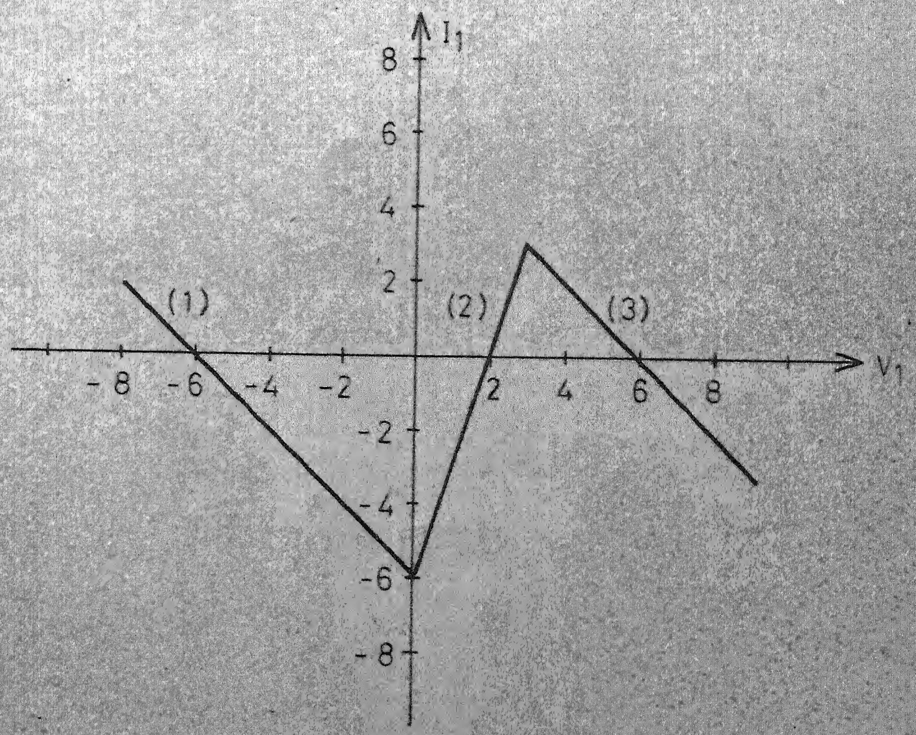

Figure I.4 Piecewise linear characterstic of resistor R_1.

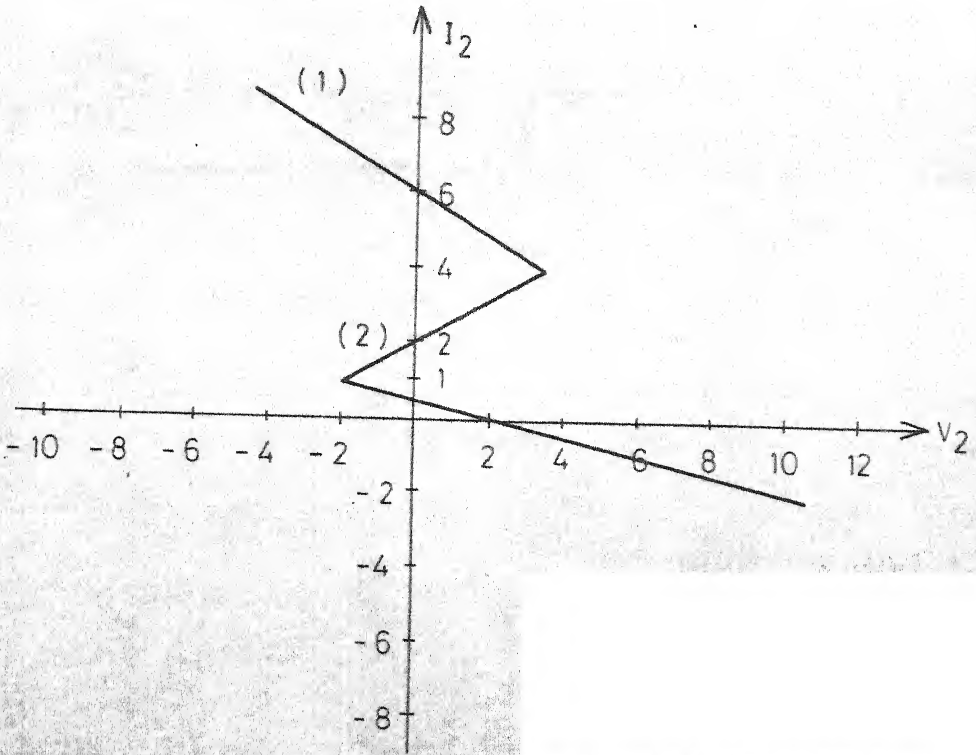

Figure I.5 Piecewise linear characterstic of resistor R_2

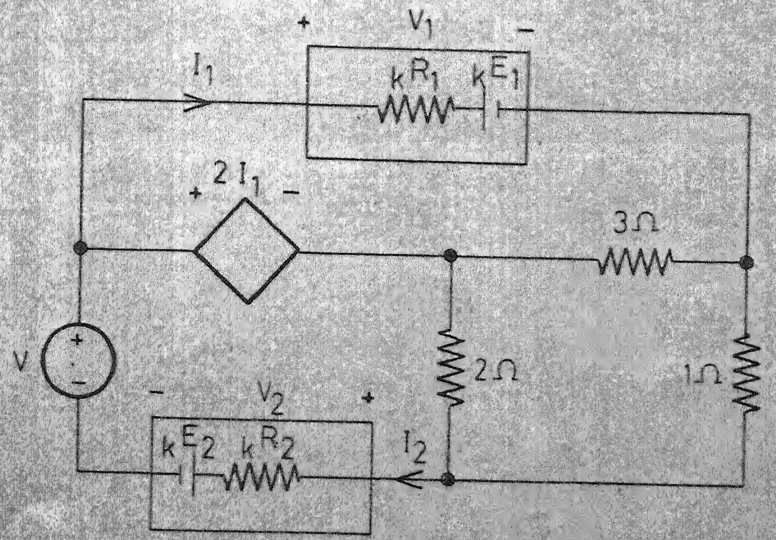

Figure I.6 Equivalent circuit of figure I.3.

TABLE I.1

MODEL PARAMETERS FOR R_1 and R_2

Resistor R_j	Segment k	${}_kR_j$	${}_kE_j$	Interval of Definition
R_1	1	-1	-6	${}_1D_1(i) = (-6,\infty)$ ${}_1D_1(v) = (-\infty,0)$
	2	$\frac{1}{3}$	2	${}_2D_1(i) = (-6,3)$ ${}_2D_1(v) = (0,3)$
	3	-1	6	${}_3D_1(i) = (-\infty,3)$ ${}_3D_1(v) = (3,\infty)$
R_2	1	-2	12	${}_1D_2(i) = (4,\infty)$ ${}_1D_2(v) = (-\infty,4)$
	2	2	-4	${}_2D_2(i) = (1,4)$ ${}_2D_2(v) = (-2,4)$
	3	-4	2	${}_3D_2(i) = (-\infty,1)$ ${}_3D_2(v) = (-2,\infty)$

$$I_1 = \frac{V - (-\frac{4}{3} + K^{R_2}) K^{E_1} - K^{E_2}}{\frac{1}{3} + \frac{4}{3} K^{R_1} - \frac{1}{2} K^{R_2} + K^{R_1} K^{R_2}} \qquad (I.4)$$

$$I_2 = [(- \frac{1}{2} + K^{R_1}) I_1 + K^{E_1}] \qquad (I.5)$$

For different segment combinations we will get different values of I_1 and I_2. Three different segments of each resistor R_1 and R_2 give 9 combinations. Use of numerical program requires nine computer runs while by symbolic network analysis program it can be solved in one computer run and gives I_1 and I_2 as equations I.4 and I.5 respectively. Then current at different values of E_1, E_2 , R_1 and R_2 can be solved, which is a much simpler task.

I.3 Extension of Symbolic Network Analysis Program (SNAP)

In order to develop a facility for obtaining symbolic network functions for complicated networks with the help of a computer, the program developed by Dr. P.M.Lin at the Purdue University was used. Though a copy of this program was available, it could not be directly used, since certain changes had to be made in accordance with the requirements of the computing facility at I.I.T. Kanpur.

Certain additional facilities are incorporated into the program to make it more versatile. These facilities are multiinput, multioutput and frequency response plotting.

IBM 1800 version of SNAP has also been developed so that the users can run their programs themselves and can do modification in data cards wherever required. Certain changes

had to be done to accommodate SNAP in IBM 1800. New facilities of handling multiinputs, multioutputs and frequency response plotting have been incorporated here also. Large scale sensitivity analysis is also added to the IBM 1800 version of SNAP.

In multiinput, one can have more than one independent source in the network. It is generally useful for the analysis of multiport networks. For example the two port parameter matrices of a given network can be calculated.

In multioutput more than one network function can be solved at a time. For example if we take I_1 as input and, I_2 and V_1 as outputs in one computer run, we can find out $\frac{I_2}{I_1}$ and $\frac{V_1}{I_1}$ etc., that is current gain as well as input impedance. It is a more economical procedure and it also saves users time.

In frequency response plotting we get the frequency response of the network. In large scale sensitivity analysis, sensitivity of the network functions with respect to different parameters, when parameter changes are large, can be calculated.

APPENDIX A

SUGGESTION FOR REDUCING ROUND-OFF ERROR

This program can further be modified to take care of the reduction of round-off error for better result. The round-off error can be reduced by storing all the coefficients of a particular term in an array and then ordering them either in an accending order or in a decending order. Then this array is added. This procedure gives the result to a better accuracy than the earlier one.

For example, let the coefficients of a term be $1 + 10^8 - 10^8 - 1$. Let the machine allow seven significant digits. 1 and 10^8 are stored as

$$1 = .1000000E{+}01$$

$$10^8 = .1000000E{+}09$$

Then $1 + 10^8$ will results in

.1000000E+09.

Thus $1 + 10^8 - 10^8$ results in zero

Now $1 + 10^8 - 10^8 - 1$ will result -1 i.e.

-.1000000E+01

which is not correct. But if these coefficient are ordered in an accending order as $1 - 1 + 10^8 - 10^8$ then the result will be zero, which is correct.

In some cases the round-off error can be reduced by double precision arithmetic. In double precision arithmetic the number of significant digits are twice as many as in the case of single precision arithmetic.

The ordering of the array is more reliable than the use of double precision arithmetic, because the round-off error in case of double precision arithmetic does not reduce under all circumstances. This depends on the type of the problem and the number of significant digits of the machine, whereas the ordering of the array gives the result correct upto the number of significant digits of the machine.

REFERENCES

1. E.V.Sorensen, 'A preliminary note on the Analytical Network Program (ANP1)' Technical report LKT 23, University of Denmark, Oct. 30, 1967.
2. L.P.Mc.Namee, H.Potash, 'A user's guide and programmer's manual for NASAP' Department of Engineering, University of California, Los Angeles, August 1968.
3. CORNAP, 'User's Manual School of Electrical Engineering, Cornell University, 1968.
4. SNAP,'A computer program for generating symbolic network functions' by P.M.Lin and G.E.Alderson, Purdue University School of Electrical Engineering, Lafayette, Indiana.

II BRIEF DESCRIPTION OF SNAP (SYMBOLIC NETWORK ANALYSIS PROGRAM) ALGORITHM

II.1 Introduction

The network function is derived by making use of the well known Signal Flow Graph (SFG) technique [(1)]. The following formula due to Mason gives the network function from the SFG.

$$\text{Network function} = \frac{\text{output}}{\text{Input}} = \sum_{i=1}^{m} \frac{P_i \Delta_i}{\Delta}$$

where $\Delta = 1 + \sum_J (-1)^J \sum_K L_{K,J}$ is the determinant of the SFG of the network.

$L_{K,J}$ is the product of the transmittances of k^{th} set of nontouching loops of order J. An n^{th} order nontouching loop is defined as the set of n nontouching loops.

P_i is the transmittance product of the i^{th} path between the output and the input.

Δ_i is the partial determinant obtained from Δ after removal of all loops intersecting the i^{th} path between the output and the input. The SFG which is used here is 'compact SFG'.

The compact SFG is the representation of the all the cutset and loop equations. This SFG is generated from the topological structure of the network. First a tree is selected for the network, then the SFG is formed to represent all cutset and loop equations. This procedure is given in details later.

The compact SFG of the network is modified to the 'closed SFG' by adding a branch of symbolic weight 'FB' from the output to the input node. The purpose of introducing the 'closed SFG' is because all orders of nontouching loops need be found as opposed to the evaluation of Mason's formula which requires enumeration of paths as well as loops.

Let Δ_c be the determinant of the 'closed SFG'. It is then noted that since $\left\{P_i\right\}_{i=1}$ is the set of all paths from the input to the output, the loops present in the closed SFG and not present in the original compact SFG will precisely be given by $\sum_m \left\{(FB)\ P_i\right\}_{i=1}^{m}$. Since the path FB contains only the input and output nodes which in turn, are present in every path P_i, i=1,2----m, it follows that the nonintersecting loop combimations, that do not touch the loops (FB) P_i, i=1,2----m, will be precisely those combinations which do not touch the path P_i, i=1,2----m. It follows that

$$\Delta_c = (FB) \sum_{i=1}^{m} P_i \ \Delta_i + \Delta$$

Thus, the network function can be found by simply sorting the terms of the determinant of the'closed SFG'.

II.2 Formulating the SFG

II.2.1 Input Data Required

A SFG is generated by SNAP from data specifying the topological structure of the network, the input output variables and the characteristics of each network branch.

The input to the network must be a single independent source (current or voltage) and the output required must be the voltage or current associated with a network branch or the voltage between any two nodes of the network.

For example, consider the common emitter transistor amplifier as shown in figure II.1 and figure II.2 (in figure II.2 the branches and nodes are numbered). Network data for it is given by table II.1.

II.2.2 <u>Finding a Tree</u>

The formulation of compact SFG starts with the choice of a network tree. The selection of network branches to be used in the tree is made as follows.

Independent voltage sources and controlled voltage sources are the first ones included in the tree. Then come the passive RLC elements in any order. In choosing $(J+1)^{th}$ branch for the tree the undirected graph formed by J branches already selected is tested to determine whether a path exists between the two terminal nodes of the $(J+1)^{th}$ branch. If so, the branch under consideration is disqualified. If not, the $(J+1)^{th}$ branch is added to the tree. Let n be the number of nodes of the network graph. When (n-1) branches are successfully chosen by the above process, we have a tree. Selection of optimum tree is referred in the Barbay and Zobrist[2] paper.

For example, for the network of figure II.1 and II.2, if the above rule is followed, the tree selected is as shown in

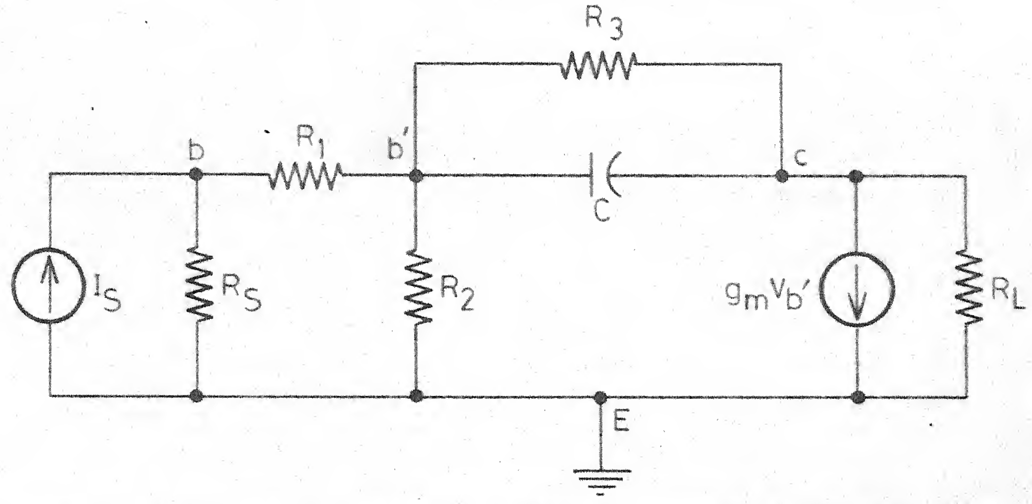

Figure II.1 Equivalent circuit of common emitter transistor amplifier stage.

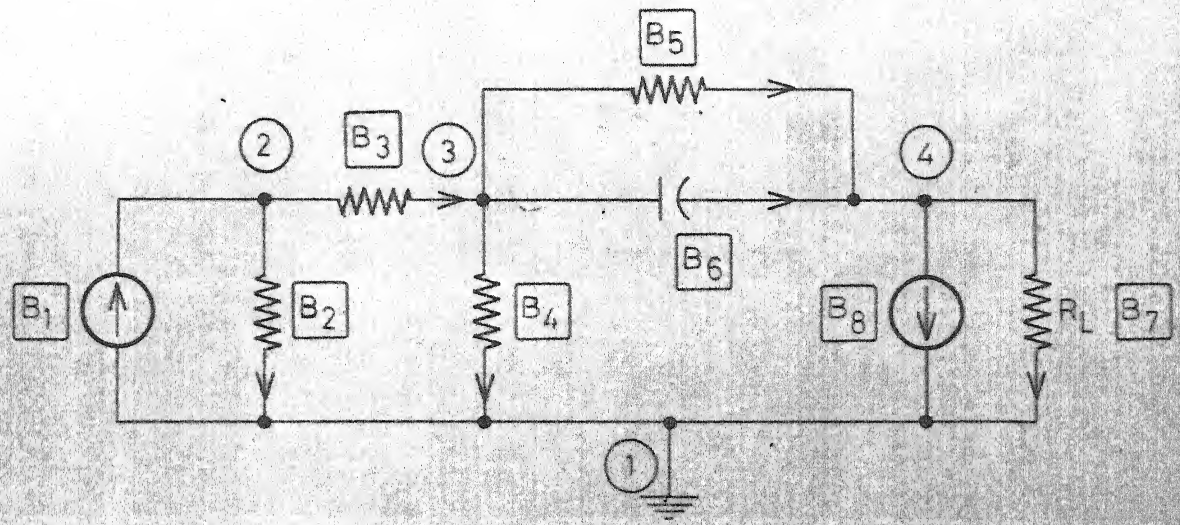

Figure II.2 Circuit of figure II.1 with the branches and nodes numbered.

TABLE II.1

NETWORK DATA

Branch Type	Branch number	Initial node	Terminal node	Symbol	Value	Control
I	1	1	2	IS		
R	2	2	1	RS		
R	3	2	3	R1	$= 1x10^2$	
R	4	3	1	R2	$= 1x10^3$	
R	5	3	4	R3	$= 4x10^3$	
C	6	3	4	CC	$= 3x10^{-12}$	
R	7	4	1	RL		
VC	8	4	1	GM	$= 5x10^{-2}$	4

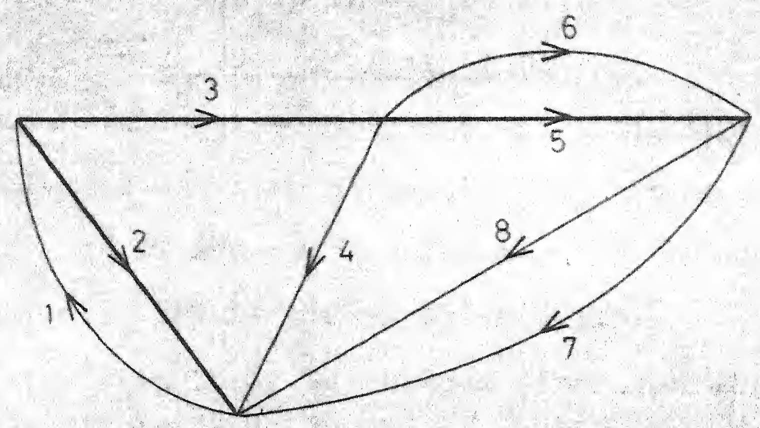

Figure II.3 Graph of the circuit of figure II.1 indicating the tree consisting of branches 2,3 and 5.

figure II.3.

The tree contains branches 2,3,5 and the **links** (those branches which are not in tree) are 1,4,6,7,8.

II.2.3 Formulation of Compact SFG

A compact SFG has node variables consisting of only tree branch voltages and link currents. Additional nodes are needed for control sources or for the output variable.

The compact SFG is generated as follows

1. For each link l_k, the unique fundamental circuit C_k containing branches b_i, i=1,2----m is found. The sets of the compact SFG branches can then be created according to the following rules.

(a) For each passive branch in the tree branch set b_i, i=1,2-----m, a directed branch in the SFG is formed from node I_{lk} to node V_{bi} with weight equal to the impedance of branch b_i, prefixed with the proper sign (Positive if the directions of l_k and b_i concur in C_k and negative otherwise).

(b) If the link l_k is a passive branch, a directed branch in the SFG is formed from each node V_{bi}, i=1,2----m to node I_{lk}, having weight equal to the admittance of link l_k, with the proper sign (negative if the directions of l_k and b_i concur in C_k , positive otherwise).

2. If any of the four types of controlled sources are present, a directed branch is created in the SFG from the controlling variable to the controlled sources having weight equal to the constant of proportionality (g_m , beta etc). If the controlling variable is a link voltage or a tree branch current, one more node is added to represent this controlling variable X (e.g. X_9 in figure II.4). X is then expressed in terms of the tree branch voltage or link current through a simple immitance relation.
3. If the desired output Y is neither a tree branch voltage nor a link current, then one node is added to the SFG to represent Y. Y is then expressed in terms of tree branch voltage or link current through a simple immitance relationship.
4. Finally, the SFG is closed by adding a branch with a symbolic weight FB, directed from the output to the input node.

For example the SFG of the common emitter transistor amplifier of figure II.1 and II.2 for the tree branches 2,3,5 is given in figure II.4 and table II.2 gives the data associated with figure II.4.

II.3 Manipulating SFG Branch Weights

From here onwards the 'compact closed' SFG will be referred as SFG. Each branch weight in the SFG is of the form (constant, symbol. s^n).

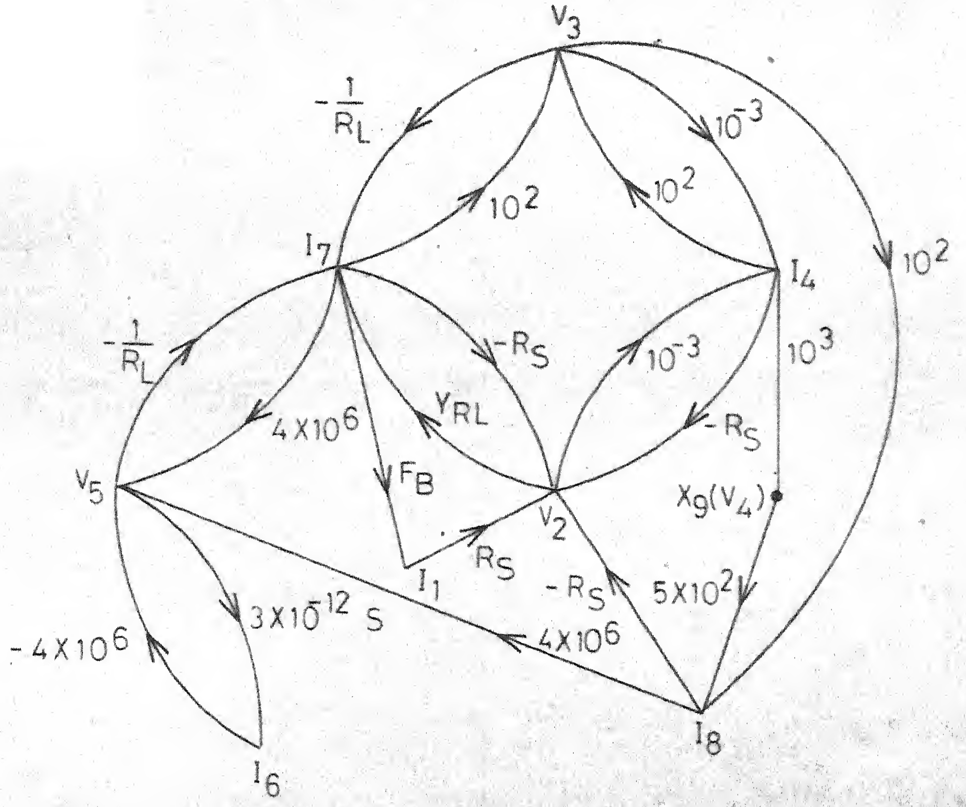

Figure II.4 SFG of the circuit of figure II.1

TABLE II.2

SFG DATA

Initial Node	Terminal Node	Exponent of S	Branch Value	Branch Symbol
7	1	0	1	FB
1	2	0	1	RS
3	4	0	$1x10^{-3}$	-
4	3	0	$1x10^{2}$	-
2	4	0	$1x10^{-3}$	-
4	2	0	-1	RS
5	6	1	$3x10^{-12}$	-
6	5	0	$4x10^{6}$	-
5	7	0	-1	$1/R_L$
7	5	0	$4x10^{6}$	-
3	7	0	-1	$1/R_L$
7	3	0	$1x10^{2}$	-
2	7	0	1	$1/R_L$
7	2	0	-1	RS
4	9	0	$1x10^{3}$	-
9	8	0	$5x10^{-2}$	-
8	5	0	$4x10^{6}$	-
8	3	0	$1x10^{2}$	-
8	2	0	-1	RS

If a branch has an initial node x_i and a final node x_f then the three parameters associated with this branch are

$C(x_i, x_f) =$ Constant

$S(x_i, x_f) =$ Symbol

$E(x_i, x_f) =$ Exponent of S.

These completely define the weight of the branch. After a loop or a set of nontouching loops has been found, it is desirable to combine the weight parameters of each branch in the loop set to form a composite loop set weight. The loop set constant may be easily formed by taking the product of the constant associated with each branch. Similarly, the loop set exponent parameter is readily formed by summing the exponent assigned to each branch. However, because computers are not particularly adept at symbol manipulation, it is inefficient with respect to both time and storage to form directly a composite loop set symbol. A much better technique is to convert each branch symbol into a numeric code. These codes are assigned as follows. Each distinct symbol, of the SFG, is stored in the array S(J) and assigned a code B^J where B is some base $\left\{ 2, 4 \text{ ------} 2^m \right\}$. Now a SFG branch having initial node x_i and final node x_f which contains the symbol S(n) will have the code

$K(x_i, x_f) = B^n$ assigned.

The real value of this coding technique stems from the fact that the composite loop set code formed by summing the codes representing the individual branch symbol can be

uniquiely decoded provided the number of identical symbols combine into any code is less than B.

For example, in figure II.4 consider the loops formed by the nodes V_3-I_7-V_3 and V_2-I_4-V_2. The weights of these loops are found as follows

For loop V_3-I_7-V_3

Loop set constnnt = $(-1) \times 10^2$

Loop set power = 0

For loop V_2-I_4-V_2

Loop set constant = $(10^{-3}) \times (-1)$

Loop set power = 0

To find loop set code, an array of distinct symbols of the SFG and their corresponding codes must be set up

Symbol Array			Code
No symbol			0
S (1)	=	FB	$\longrightarrow (4)^0$
S (2)	=	R_S	$\longrightarrow (4)^1$
S (3)	=	$1/R_L$	$\longrightarrow (4)^2$

Loop set code for loop V_3-I_7- V_3

$= K(V_3,I_7) + K(I_7,V_3)$

$= 16 + 0$

Loop set code for loop V_2-I_4-V_2

$= K(V_2,I_4) + K(I_4,V_2)$

$= 0 + 4$

So the weight of loop V_3-I_7-V_3 and V_2-I_4-V_2 are respectively

$-10^2 \times \frac{1}{R_L}$ and $-10^{-3} \times R_S$

These loops do not touch; therefore the composite weight of second order nontouching loop formed by loops V_3-I_7-V_3 and V_2-I_4-V_2 is as follows.

Composite loop set constant = $-10^2 x(-10^{-3}) = (10)^{-1}$

Composite loop set S power = 0

Composite loop set code = 16 + 4 = 20

Now to decode the loop set code 20, it can be written as $(4)^1 + (4)^2 = R_S \times \frac{1}{R_L}$ which is indeed the symbol associated with the loop immitance product. Therefore,

composite loop set weight $= (10)^{-1} \times R_S \times \frac{1}{R_L}$

Each loop set contributes to a term in the network function. As each loop set is generated and coded, it is compared with existing terms. If a term with same symbol code and power of S exists, then constant of the term is updated by adding to it the constant of the new loop set otherwise a new term is created. This process of coding and decoding of symbols is an important step towards reducing the storage requirements.

After all loop sets have been found, the transfer function is complete and it remains only to transform the symbol code of each term into its corresponding symbol set.

II.4 Generating First Order Loops

Let the nodes of the SFG be labelled 1,2-----N. All first order loops which contain node J (J=1 initially) can be found by conceptually splitting node J into two nodes, one node containing all incoming branches and the other containing all

outgoing branches and then enumerating all paths between these two nodes. All branches going into node J are then removed and the process is repeated for node J+1. This procedure will produce all circuits with no duplication.

Consider the SFG in figure II.4. The topological structure of the SFG can be completely described by a routing table (table II.3), where the entries in the J^{th} row are the set of all nodes of distance one from node J. The entries of each row are made to decrease as the column subscript m increases. This routing table is used for obtaining all circuits in the SFG. Since the rows are arranged in the order of the numbering of the nodes, once all the circuits through a given node have been found, that row will be eliminated while obtaining the circuit through the rest of the nodes. In addition whenever the number of the node already considered appears as the right most entry , that entry is also eliminated. This procedure eliminates duplication of circuits. As an example in using the routing table II.3, the following four circuits can easily be shown to form the complete set of circuits through the node 1.

1 - 2 - 7 - 1

1 - 2 - 4 - 9 - 8 - 5 - 7 - 1

1 - 2 - 4 - 9 - 8 - 3 - 7 - 1

1 - 2 - 4 - 3 - 7 - 1

While finding circuit through a node J, loops formed out of the nodes appearing in these circuits should be avoided.

TABLE II.3

ROUTING TABLE

$$
R(J,M) \;=\; \begin{array}{c|cccc}
1 & 2 & & & \\
2 & 7 & 4 & & \\
3 & 7 & 4 & & \\
4 & 9 & 3 & 2 & \\
5 & 7 & 6 & & \\
6 & 5 & & & \\
7 & 5 & 3 & 2 & 1 \\
8 & 5 & 3 & 2 & \\
9 & 8 & & &
\end{array}
$$

For example, while finding the circuit through node 1, if we proceed as 1 to 2, 2 to 7, 7 to 5, 5 to 6 and 6 to 5, we see that nodes 5 and 6 form a loop. So the path is retraced to node 7 and from here we proceed to node 3. Again we find a loop. So finally we go back to node 1 from 7. Thus we get a circuit 1-2-7-1. In order to find the existance of a loop instead of comparing the prospective node to each node already included in the path, it is much more efficient to define the binary sequence S of length equal to the number of nodes in the SFG as follows

$$S = X(N)\ X(N-1)\ \text{---------}\ X(1)$$ where N is the number of nodes in the SFG

Function X is defined as

$$X(I) = \begin{cases} 1 & \text{if } I \text{ is contained in the path node sequence} \\ 0 & \text{if } I \text{ is not contained in the path node sequence} \end{cases}$$

Another binary sequence for the node (J) under consideration is defined as

$$S(J) = X(N)\ \text{---------}\ X(1)$$

where

$$X(I) = \begin{cases} 0 & \text{for } I \neq J \\ 1 & \text{for } I = J \end{cases}$$

To see whether node J is present in the path node sequence an 'AND' operation is performed between sequences S and S(J); if the result is zero then node J is not present in the path node sequence S, otherwise it is present.

For example, in the above example, if we proceed as 1 to 2, 2 to 7, 7 to 5, 5 to 6, then path node sequence considered till now is 1-2-7-5-6 which gives

S = 001110011

and to find next node of the path if we proceed as 6 to 5, then node under consideration is 5, then

S(5) = 000010000

and to see whether this node is qualified node or not we perform 'AND' operation as

S. AND. S(J) = 000010000 $\neq$ 0

shows that node 5 is already present in the path node sequence and it will form a loop. Therefore this node is disqualified.

Additional insight may be obtained by viewing the path finding technique graphically . That is the process by which paths generated can be observed by applying the following two rules directly to the SFG

(1) Let node J be the last node added to the path node sequence (initially J = input node). To select the next node ,traverse through that branch, connected to node J that goes to the highest numbered node satisfying both the following requirements

(a) We did not back up from this node while applying rule 2 and

(b) This node is not included in path node sequence.

Repeat this process untill the output node is reached , then store the node sequence and go to rule (2) or until no

new node can be found to satisfy (a) and (b) (then go to rule (2)).

(2) Back up along the path just found (this is always possible unless we are at the input node in which case all paths have been found) until a new route can be taken according to rule 1.

For example, the heavy lines of figure II.5 show the circuit which results from applying rule 1 when circuit through node 1 is considered. Generating a second circuit requires backtracking to node 8 , then continuing the sequence 3-7-1. The graphical technique for listing all paths can be helpful when solving problems by hand.

II.5 Generating Nontouching Loops of Order Two or More

This part will generally require the most time unless the network contains many distinct symbols. It is therefore necessary to exercise considerable care in developing an algorithm for finding all orders of nontouching loops.

In general to find loop sets of all orders, some comparison between the node sequences of the different loops must be made. A brute force technique is simply to store all the node sequences of the first-order loops and to find nontouching loops by direct comparison of the nodes contained in the loop. Of course, storage is also needed to indicate the loops contained in some of the higher order combination, but this storage is necessary even in more efficient techniques which follow.

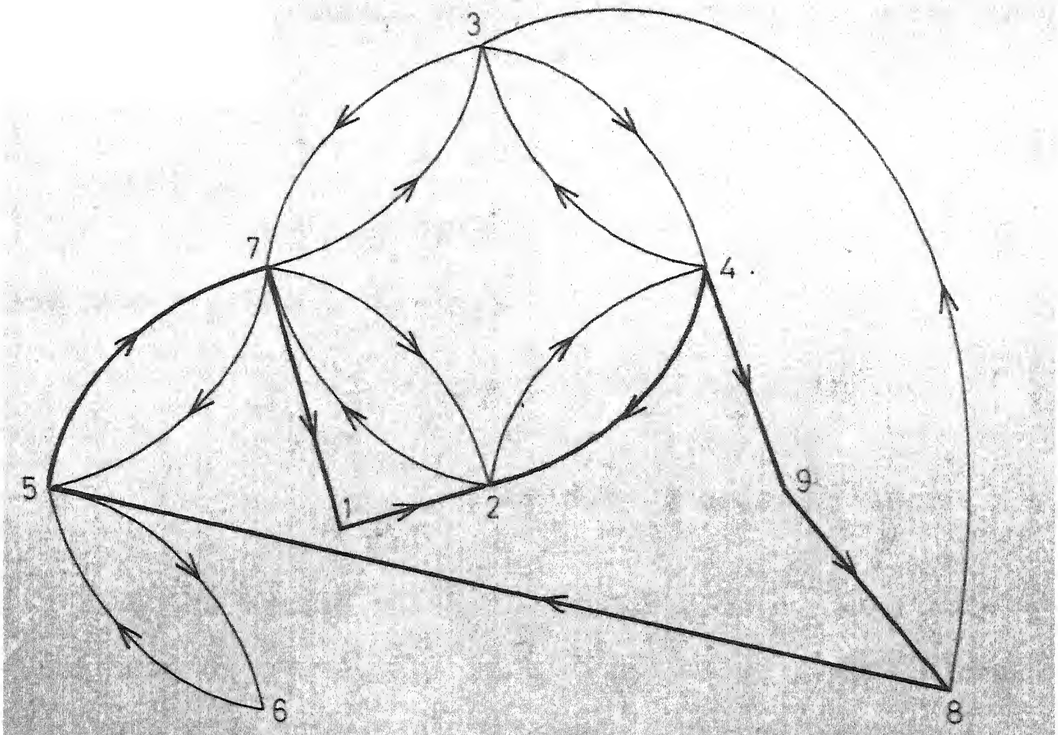

Figure II.5 SFG indicating the closed path including the nodes 1,2,4,9,8,5 and 7

The above method is improved considerably if instead of directly comparing the nodes of loops A and B to determine if they touch, a binary array F(I), associated with each first order loop, of length equal to the number of nodes in the SFG is defined as

$$F(I) = \begin{cases} 1 & I \in \{\text{nodes in loop } A\} \\ 0 & \text{othersise} \end{cases}$$

and then tested as follows

$$\text{IF} \quad F(J) = \begin{cases} 0 & \text{all } J \in \{\text{nodes in loop } B\} \Rightarrow \text{loops do not touch} \\ 1 & \text{any } J \in \{\text{nodes in loop } B\} \Rightarrow \text{loops touch} \end{cases}$$

In the method which is actually used here, only a single code need be stored for each first order loop instead of the complete node sequence. As each first order loop is generated, it is assigned an integer code whose binary representation shows the set of nodes in the loop.

For example, if loop A contains the nodes $\{1,2,7\}$, loop B contains the nodes $\{5,6\}$ and C contains $\{1,2,4,9,8,5,7\}$, the codes are evaluated as

$$A = (001000011)_2 = (67)_{10}$$

$$B = (000110000)_2 = (48)_{10}$$

$$C = (111011011)_2 = (475)_{10}$$

To determine whether the two loops touch or not the masking operation 'AND' is used. Thus

$(A).AND.(B) = (000000000)_2 = 0$

The result is zero indicating that loops A and B do not touch.

And $(A).AND.(C) = (001000011)_2 \neq 0 = (67)_{10}$

The result is not zero indicating that loops A and C touch.

In our computer IBM 7044, 'AND' operation of logical strings is not possible. Therefore, a subroutine is written to find the 'AND' operation of binary equivalent of two decimal numbers.

REFERENCES

1 'Automatic Control Systems' By Benjamin C.Kuo, Professor of Electrical Engineering, University of Illinois, Prentice-Hall of India Private Limited, New Delhi, 1973.

2. J.E.Barbay and G.W.Zobrist,'Distinguishing characteristics of the optimum tree', 5th Allerton Conf. on circuit and system theory, pp 730-737, 1967.

3. P.M.Lin and G.E.Alderson, 'A computer program for generating symbolic network functions (SNAP)', Purdue University, School of Electrical Engineering, Lafeyette, Indiana.

III. MODIFICATION OF SNAP

III.1 Introduction

This chapter describes the implementation details and the additional facilities like multiinput and multioutput, which have been added to SNAP.

III.2 Implementation of SNAP on IBM 7044

The change that had to made to implement SNAP on IBM 7044 was in connection with a 'COMMON' statement used in the main program and subroutine SFG. This was required because the variables used in the 'COMMON' statement were real variables in the main program, but integers in the subroutine SFG, which is not allowed in IBM 7044, available at I.I.T. Kanpur. The 'COMMON' statement in the main program and in the subroutine SFG was the following

COMMON	SEMPON, SEMPOD, POLY	In main program
COMMON	NF, NS, IB	In subroutine SFG

These statements were deleted both in the main program as well as in the subroutine SFG, and this change did not affect the program. Instead these variables used in the 'COMMON' statement were defined before using them in the respective programs. This 'COMMON' statement was used in the original SNAP to save memory because corresponding common variables are stored in the same area in memory. But this facility had to be sacrifised in the modified program.

An error was detected in the original SNAP when a differential amplifier circuit containing 12 branches (which results in a total number of 59 paths and circuits, in the SFG) was solved. The dimension of the variable 'SMBOL' was not sufficient. The dimension of it should have been NBG (Number of branches of compact SFG i.e. 75) instead of NBN (number of branches in the network i.e. 25) because its subscript takes values up to NBG.
(Note: Refer to Appendix A for one more error)

III.3 Multioutput Facility

This facility is used to obtain more than one output function in a single computer run. For this the following technique is used.

Augment the original network by appending at one end a series connection of dependent voltage sources to the given network such that,

(a) to each branch current I_J desired as an output, there corresponds a dependent voltage source which depends on I_J and has symbolic weight AAA-----etc. and

(b) to each voltage $V_{0\,AB}$ desired as an output, there corresponds a set of dependent voltage sources each dependent upon a voltage across one of the branches in the path between A and B and all having symbolic weight BBB --- etc.

By specifying the output to be the voltage across the entire series connection of dependent voltage sources, we get an output function.

From this function, the output function corresponding to I_J and V_{OAB} can be obtained by taking into account only those terms which are coefficients of AAA as corresponding to the output I_J and those which are coefficients of BBB as those corresponding to V_{OAB}.

For example, figure III.1 illustrates the network augmentation needed to find voltage V_1 and V_7 across R_1 and R_L respectively, for the given common emitter transistor amplifier shown in figure II.1 and II.2.

We get an output function as follows

$$\text{Output function} = \frac{V_{46}}{I_S} = AAA.P_1 + BBB.\ P_2$$

Then the network functions $\frac{V_2}{I_S}$ and $\frac{V_7}{I_S}$ are given by

$$\frac{V_2}{I_S} = P_1 \text{ and } \frac{V_7}{I_S} = P_2,$$

P_1 and P_2 are polynomials containing the circuit symbols

Detailed Algorithm

Corresponding to each required current through a branch J, a current controlled voltage source is added to the network. The controlling variable is the current in the branch J. The symbol associated with the branch is given in the 'DATA' statement. Similarly, if the output is a voltage across a branch J, then a voltage controlled voltage source is added to the network with the controlling variable as the voltage in the branch J. The symbol associated with this branch is given in

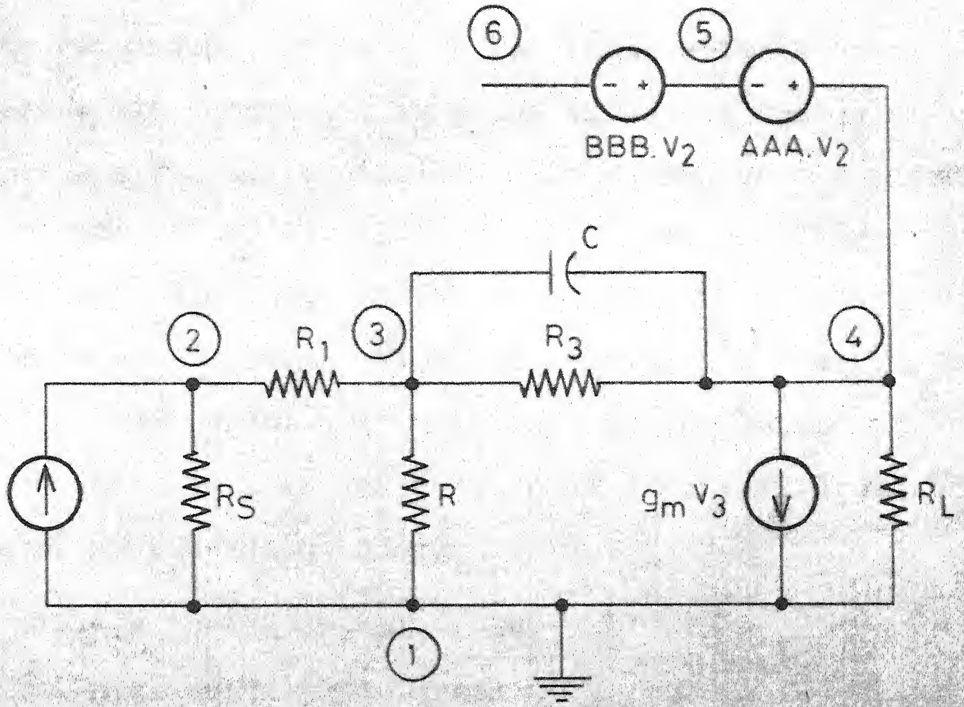

Figure III.1 Augmented network for multioutput.

the 'DATA' statement which is different from those for other controlled sources. If the required output voltage is a voltage across two nodes, across which a single network branch is not connected but a path can be found through a number of tree branches, a voltage controlled voltage source is added for each branch (J) of the path having the controlling variable at that branch (J). All of such controlled sources will have same symbol defined in the 'DATA' statement. Therefore the number of such symbols used is same as the number of outputs.

For example , in the network of figure II.1 and II.2, the required output voltages are

1. Voltage across branch 2, and
2. Voltage across nodes 2 and 4 (V_{24})

For voltage V_{24} the path between nodes 2 and 4 is found out through the tree (consisting of branches 2,3 and 5). The path between node 2 and 4 contains branches 3 and 5. Therefore, as shown in figure III.2, we have two augmented voltage dependent source branches B_{10} and B_{11} with the symbols BBB multiplying the voltages V_3 and V_5 respectively.

III.4 Multiinput Facility

Program SNAP permits only one independent source. However, this program has been modified by the following technique in order to be able to handle more than one independent source.

Let W_i, i=1,2 -----n represent a set of n independent sources either voltage or current. Let W_1 be the permitted independent source and let W_2, W_3 ----W_n be the sources which

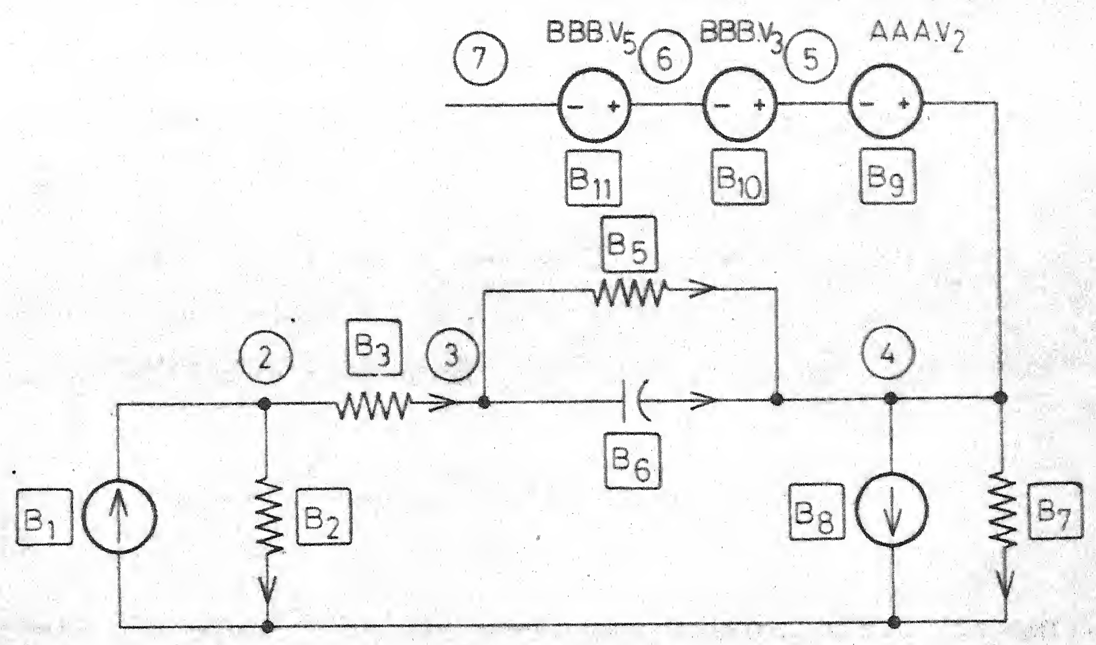

Figure III.2 Modified network for multioutput.

are dependent on W_1 with proportionality factors as shown below

$$K_1 = \frac{W_2}{W_1} \; ; \; K_2 = \frac{W_3}{W_1} \; ; \; - - - - - K_{n-1} = \frac{W_n}{W_1}$$

The output polynomial can be written as

$$\frac{W_{out}}{W_1} = \frac{P_1 + P_2 K_1 + P_3 K_2 + - - - - P_n K_{n-1}}{\Delta}$$

where Δ and P_i, i=1,2 -----n are polynomials. The output function can then be written as

$$W_{out} = \frac{P_1 W_1 + P_2 W_2 + P_3 W_3 + - - - P_N W_N}{\Delta}$$

Network functions can be obtained as follows

$$\frac{W_{out}}{W_1} = P_1 \; ; \; \frac{W_{out}}{W_2} = P_2 \; ; \; - - - - - - \frac{W_{out}}{W_N} = P_N$$

Detailed Algorithm

If the number of input sources in the network are more than one, then the first one is taken as an independent source and the rest are treated as the sources dependent on the first one with the proportionality factor such as K_1, K_2 - - etc. If the first independent source is a current source then the other dependent sources will be current controlled. If the dependent sources are current sources then we have current controlled current sources and if these are voltage sources then we have controlled voltage sources. Similarly if the

first source is a voltage source then the rest of the dependent sources will be voltage controlled and these will be voltage controlled voltage sources if the dependent sources are voltage sources; if the dependent source is a current source then it will be a voltage controlled current source. In all these controlled sources the controlling variable will be the independent source. All the independent sources except the first one are replaced by the corresponding controlled sources.

For example, the common emitter transistor amplifier having two independent sources I_S and I_L is shown in figure III.3 and the modified circuit is given by figure III.4.

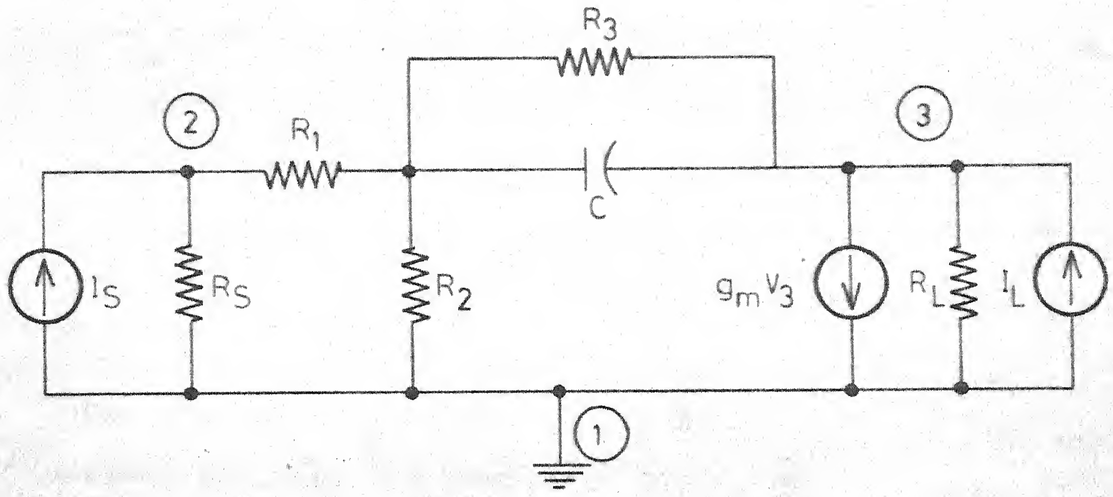

Figure III.3 Multiinput equivalent circuit of common emitter transistor amlifier stage

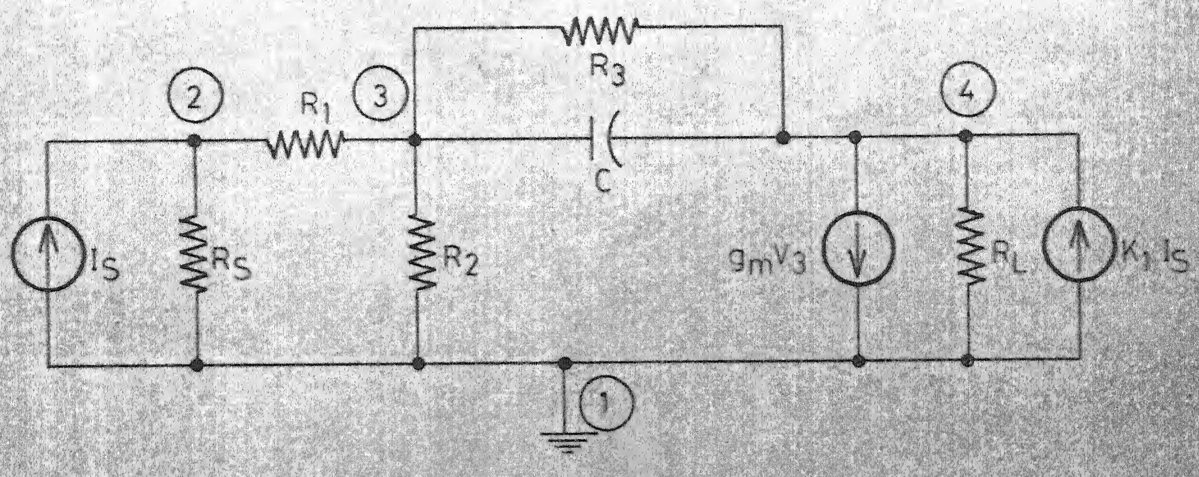

Figure III.4 Modification of circuit of figure III.3 for multiinput

APPENDIX A

An error was encounted when developing SFG for voltage controlled current sources. If the controlling branch is a link and the type of the link is either resistance (R) or inductance (L) or impedance (Z) than for creation of extra node of link voltage from link current, the symbol of the link should not be inverted. In the subroutine SFG of the 7044 version and in the subroutine SUB2 of the 1800 version of the program, the statement 209 is changed from

```
209            KONSO (LIST) = 1
```

to

```
209            KONSO (LIST) = KUNO
```

IV. IBM 1800 VERSION OF SNAP

IV.1 Introduction

This chapter describes the implementation details of SNAP on IBM 1800.

IV.2 Changes Made for Converting IBM 7044 Version of SNAP to IBM 1800 Version of SNAP

1. All the six character variable names are changed into five character names. Generally the last character of the six character variable name is deleted except in a few cases. For details see appendix A.

2. All logical 'IF' statements are changed into arithmetic 'IF' statements. For example statements

```
      IF (I.EQ.J) GO TO 15
      I = I+1
```

is changed into statements

```
      IF (I-J) 10, 15, 10
   10 I = I+1
```

3. In the write statements, the statement that is to be printed, is kept within quote marks (') instead of stars (*)

For example

```
          WRITE (6,10) NOD
      10  FORMAT (1X, * NUMBER OF NODES = * , I3 )
```

Statement number '10' is changed as follows

```
10  FORMAT (1X, 'NUMBER OF NODES = ' , I3)
```

4. Dimensions are reduced because the dimensions of the variables, in the IBM 7044 version of SNAP are very large, they cannot be accommodated in IBM 1800. For details see appendix B.

5. The program is divided into two parts because the whole program cannot be accommodated in the memory; these parts are executed sequentially by using the 'LINK' subroutine. However such parts are stored using a *STORECI card prior to execution. For example, let the names of the parts be PART1 and PART2, Let PART1 be executed first and then PART2. Then PART1 will contain two additional cards, one in the beginning and one in the last as follows

```
EXTERNAL   PART2
   PART1  program

CALL   LINK (PART2)
END
```

When PART1 is executed, PART2 is executed automatically The variables from first part to second part are transferred by 'COMMON' statements.

6. The memory of IBM 1800 is insufficient to accommodate each one of these parts, so each part is further divided into number of subroutines. All the subroutines are not loaded simultaneously. Only those subroutines are loaded which are executed at the same time. This is done by the statement 'LOCAL'. The function of this statement is to call a subroutine into the memory when it is required. The restriction is that no subroutine in 'LOCAL' should call another subroutine, but in this program one subroutine calls another subroutine, which is done by 'Group LOCAL' ; in 'Group LOCAL' a local subroutine can call another local subroutine provided they are in one group. It is expressed as follows.
For example

```
LOCAL   (SUB1, SUB2), SUB3, (SUB4, SUB5, SUB6)
```

In this SUB1 can call SUB2 but cannot call any other local subroutine such as SUB3, SUB4 etc.

7. 'BLOCK COMMON' facility is not available on IBM 1800. Therefore the variables are transferred by argument list, from one subroutine to another subroutine or from main program to subroutine and vice-versa instead of 'BLOCK COMMON'.

8. In 'DATA' statement H-format cannot be used. That's why whatever is in 'DATA' statement, is included within quote marks (').

For example

DATA FB, SB/ 3H FB, 3H 1 /

is changed into

DATA FB, SB/' FB', ' 1 '/

9. IBM 1800 typewriter prints the character '#' instead of ' = ' character. If a correct output is to be printed from the IBM 1800 typewriter, the 8-6 combination keys on the IBM 29 key punch machine must be used. In data cards instead of ' = ' character. 8-6 combination key on the IBM 29 key punch machine (which is numeric V) is to be punched.

10. ' ONE WORD INTEGERS' control card is included. Otherwise integers will take two words in memory.

IV.3 <u>Important Considerations for Conversion from IBM 7044 to IBM 1800 Version of SNAP. The following are the Points which are to be carefully considered</u> .

1. Program is divided into two parts in such a way, that 'COMMON' block is as small as possible because of small memory of IBM 1800.

2. While dividing the program into parts and parts into subroutines care should be taken for back and forth referencing.

3. Dimension of the variables are reduced in proportion.

4. We should generally use base for symbol code as 4 because if base for symbol code is used as 8 then we cannot use more than five symbols (including multioutput symbols as AAA, BBB - - - etc. and multiinput symbols such as K1,K2 - - - etc.) because if a sixth symbol is used, the code for that will be 8^5 which is equal to 32768 (2^{15}) which is greater than the maximum integer ($2^{15}- 1$) represented in IBM 1800 . If one wants to use more than five symbols, base for symbol code should be 4 or less.

5. In 'Group Local', subroutine of the one local group cannot call a subroutine of the other local group. If any subroutine 'S' is called by the subroutines of the different groups then subroutine 'S' should be stored by different names and one of these names should be in each local group.

APPENDIX A

List of variables which are changed while converting IBM 7044 version of SNAP to IBM 1800 version of SNAP.

Original Variables	Changed Variables
DECODE	DECOD
INTREE	INTEE
IPOWER	IPOWE
IQUALX	IQUAX
KAPMAX	KAPMA
KBASIS	KBASI
NFIRST	NFIRS
NOCTOT	NOCTO
NOTREE	NOTRE
NPCODE	NPCOD
SEMBOL	SEMBL
SIMBOD	SIMBD
SIMBON	SIMBN
SYMBUL	SYMBU
TCONS2	TCONS
TCONSG	TCONG

APPENDIX B

The dimensions of the subscripted variables are function of variables NBN, NBG, NPAC, NTO, NSPT, NEXPS, NRI, NCI, NRS, NEON, which are defined as follows and their original and changed values are also given.

		Original value	Changed value
NBN	Number of network branches	25	15
NBG	Number of branches in SFG	75	30
NPAC	Number of paths plus circuits	220	125
NTO	Number of terms in output	125	40
NSPT	Number of symbols per term in output	16	8
NEXPS	Number of different powers of S	12	5
NRI	Maximum number of nontouching loops	12	8
NCI	Maximum number of loops nontouching any given loop	75	40
NRS	Number of repeated symbols	9	9
NEON	Number of nontouching pairs of loop	900	400

REFERENCE

1. IBM 1800 USERS' MANUAL, By M.V.Rao and Dr.S.C.Mehta, Department of Chemical Engineering, Computer Centre, Indian Institute of Technology, Kanpur, August 1973.

V. FREQUENCY RESPONSE PLOTTING FACILITY AND LARGE SCALE SENSITIVITY ANALYSIS

V.1 Introduction

This chapter describes the details of frequency response plotting for different set of values of symbols. From this plotting facility large scale sensitivity of the network function with respect to these symbols can be calculated.

V.2 Frequency Response Plotting

The expression for network function contains symbols in both numerator and denominator. After substituting the values of all the symbols, the numerator and denominator are split up into real and imaginary parts.

$$N(\omega) = \frac{N}{D} = \frac{R_N + j\omega I_N}{R_D + j\omega I_D}$$

where R_N = Real part of numerator

I_N = Imaginary part of numerator

R_D = Real part of denominator

I_D = Imaginary part of denominator

From this expression the magnitude of network function is calculated as

$$|N(\omega)| = \sqrt{\frac{R_N^2 + \omega^2 I_N^2}{R_D^2 + \omega^2 I_D^2}}$$

The phase angle is calculated as follows.

$$N(\omega) = \frac{(R_N + j\omega I_N)}{(R_D + j\omega I_D)} \times \frac{R_D - j\omega I_D}{R_D - j\omega I_D}$$

$$= \frac{(R_N R_D + \omega^2 I_N I_D) + j\omega (I_N R_D - R_N I_D)}{R_D^2 + \omega^2 \quad I_D^2}$$

The phase angle,

$$\varphi(\omega) = \tan^{-1} \frac{\omega (I_N R_D - R_N I_D)}{(R_N R_D + \omega^2 \quad I_N I_D)}$$

Both the magnitude and the phase angle are the functions of frequency (ω). They are plotted against frequency.

In frequency response plotting, the frequency scale is a log scale. The magnitude or phase angle are in linear scale. The scale of these variables is given by maximum and minimum value of these variables.

All the symbols of network function are stored in an array. Identical symbols are sorted out and their corresponding values are stored in temporary locations. After calculating the magnitude and phase angle of the network function for each set, the symbol values which are stored in temporary locations are replaced by new set of symbol values. ’ter this the magnitude and phase angle for different set of ılues of symbols can be calculated.

V.3 Large Scale Sensitivity Analysis

The large scale sensitivity of the network function with respect to any of the symbols can be calculated by plotting the frequency response for different values of that particular symbol keeping rest of the symbols constant.

APPENDIX A

A BRIEF LIST OF LIMITATIONS ON THE SIZE AND TYPE OF NETWORK ALLOWED

	IN IBM 7044	IN IBM 1800
Number of Network Branches (NBN)	25	15
Remark SNAP cannot handle all networks having NBN branches or less. Other factors such as SFG characteristics (number of higher order loops for example) and number of network symbols to name a few can further limit the size of the network.		
Maximum Number of Elements that can be Represented by the same symbol (k)	7	3
Remark This number can be increased to 2^n-1 by increasing the symbol code base used to 2^n, $n > \log_2 (k+1)$ on the input data card 2.		
Number of Different Powers of S (NEXPS)	12	5
Remark Sufficient for network containing no more than NEXPS reactive elements		

	IN IBM 7044	IN IBM 1800
Estimate of the Maximum Number of Distinct Network Symbols (Including Multiinput and Multioutput Symbols) Permitted (SYM)	11	7

Remark This restriction results from the fact that SNAP can contain no more than 125 different symbol combinations in the output in IBM 7044 and 40 different symbol combinations in the output in IBM 1800.

APPENDIX B

USER'S MANUAL

INFORMATION NEEDED BY USER

Program SNAP (Symbolic Network Analysis Program)

Purpose To obtain the network functions $\frac{V_{out}}{V_{in}}$, $\frac{V_{out}}{I_{in}}$, $\frac{I_{out}}{V_{in}}$ or $\frac{I_{out}}{I_{in}}$ as a ratio of two polynomials of the following type:

(1) All network element values are represented by symbols (the symbols need not all be different)

Examples:

$$\frac{V_{out}}{V_{in}} = \frac{S^2LRC}{S^2 2LRC + S(L+R^2C) + R}$$

$$\frac{V_{out}}{I_{in}} = \frac{ZYR^2}{2ZYR + Z + R^2Y + R}$$

(2) Some element values are specified numerically, some symbolically,

Example:

$$\frac{V_{out}}{V_{in}} = \frac{S^2}{S^2 2R + S(.5x10^6+150R^2)+.75x10^8R}$$

(3) All element values are given numerically,

Example:

$$\frac{V_{out}}{V_{in}} = \frac{S^2}{2S^2 + 2x10^4S + .75x10^8}$$

Description Program SNAP is designed to handle lumped, linear, time invariant networks containing the following type components

(1) Two terminal circuit elements-resistance, inductance and capacitance.

(2) Two terminal network described by an admittance or impedance parameter.

(3) All types of controlled sources.

(4) Independent sources.

(NOTE : Mutual inductance, ideal transformers gyrators, etc. can be modeled with elements in (1) and (3)).

Network Data Required: After the network components has been modeled by the type elements allowed, the branches and nodes are to be numbered consecutively starting with 1 and reference directions for each branch current are to be chosen. The following gives the sequence of data cards needed to describe the network.

DATA REQUIRED IN IBM 7044

CARD 1

Columns	Contents
1-72	Title card (all 72 columns are reproduced in the output. Column 1 should not be blank).

CARD 2

Columns	Contents
1-5 (right adjusted)	Number of nodes in the network
6-10 (right adjusted)	Number of branches in the network

The following three entries are optional

11-15 (right adjusted)	Numb er base of symbol codes (automatically set to 8 if left blank)
21	1 if a description of the SFG is to be listed, blank otherwise
22	1 if all loops (circuits) in the SFG are to be listed (node sequence), blnak otherwise

CARD 3

Columns	Contents
1-5 (right adjusted)	Number of input sources
6-10 (right adjusted)	Number of outputs

CARD 4 thru (b+3)

(b = number of network branches)

Note 1: Each card describes one network branch (element).

Note 2: If output is a voltage (current) associated with a particular branch, then the data card describing this branch should be ent·red first (last) among

the branch data cards (cards 4 thru (b+3)) to insure that this branch will be chosen as part of the tree (cotree).

Note 3: When a large number of branches share one common terminal, it is better to place these branches first starting with card 4 (card 5 if note 2 applies).

Columns	Contents		
1-2 (left adjusted)	Element type;	E:	voltage source
		I:	current source
		G:	Conductance
		R:	resistance
		L:	inductance
		C:	capacitance
		Z:	impedance
		Y:	admittance
		CC:	current controlled current source
		CV:	current controlled voltage source
		VC:	voltage controlled current source
		VV:	voltage controlled voltage source

Continued Columns	Contents
3-5 (right adjusted)	Element number- - all elements of the network must be assigned a distinct number (positive integer). For greatest efficiency, the numbering should be consecutive.
6-10 (right adjusted)	Initial node - -this is relative to the arbitrarily chosen current direction.
11-15 (right adjusted)	Terminal node- -this is relative to the arbitrarily chosen current direction.
17-19 (right adjusted)	Element symbol- -the element's value, if not specified, is represented by this symbol.
20	Equal sign (=) if element is to be assigned a value. Leave blank if element value is to be represented in symbolic form.
21-32 (right adjusted)	Element value (if known)- -Format is E12.5. Units should be compatible with element type as specified in columns 1-2; for example,R is expressed in ohms, G in mhos.
33-35 (right adjusted)	If element is a dependent source, enter the element number of its control.

If n is the number of input sources then n cards will give input sources each containing one input source. These cards are as follows

Columns	Contents
1-5 (right adjusted)	Branch number of source
10	1 if input is current source otherwise blank

So far the number of cards punched is equal to $(b+3+n)$.

If m is the number of outputs cards $(b+3+n+1)^{th}$ to $(b+3+n+m)^{th}$ are punched as follows

Columns	Contents
1-5 (right adjusted)	Network branch number associated with output (leave blank if output is voltage across more than one branch).
5-10 (right adjusted)	Node number corresponding to positive output voltage node (leave blank if columns 1-5 are not blank).
10-15 (right adjusted)	Node number corresponding to the negative output voltage node (columns can be left blank if 1-5 are not blank).
20	1 if output is current through a branch, blank otherwise (when columns 5-10 and 10-15 are not blank this should be blank).

Data Required in IBM 1800

If the program is stored in IBM 1800 disk. The first two control cards are as follows

```
// JOB
// XEQ SNAP          F X
       8            16 17
```

The rest of the data cards are same as the data cards in IBM 7044.

The important difference is that instead of '=' character, code 8-6 should be punched. This is the IBM 1800 typewriter equivalent of '=' character (code 8-6 is numeric V in the IBM 29 key punch machine).

In 'Card 2' if columns 11-15 are left blank the base for symbol code will be automatically 4 instead of 8.

If frequency response plotting is required punch '1' in the column 1 of next card, otherwise use a blank card.

Frequency Response Plotting Data

After the earlier data cards the first card contains

Column	Contents
1-10 (right adjusted)	Number of sets of symbols.
10-20 (right adjusted)	Number of frequencies to which plotting is done.

The next set of cards contain the values of symbols in E 12.5 format (starting from frist column). These values are fed after seeing the printout as follows.

Network function contains symbol. The machine will print the first symbol and then wait for its values. The user has to give its value, and then machine will print the second symbol and wait for its value. The user has to give the value of this symbol. This process is repeated for all the symbols. After one set the machine will wait for rest of the sets of symbols. The user has to give the rest of the sets of symbols values simultaneously in the same order.

Note: While plotting any network function of a network containing multiinputs, using multiple outputs. The multiinput and multioutput symbol value corresponding to this network function is punched as one (in E12.5 format) and the rest of the multiinput and multioutput symbols value are punched as zero in the data cards.

Whenever the symbol occurs as '1', the user should give its value as '1' in E12.5 format.

DATA CARDS FOR COMMON EMITTER AMPLIFIER

```
*************      COMMON EMITTER TRANSISTER AMPLIFIER      **************
     4    8     4    11
     1    1
I    1    1    2  I1
R    2    2    1  RS
R    3    2    3  R1=        .1E+3
R    4    3    1  R2=        .1E+4
R    5    3    4  R3=        .4E+7
C    6    3    4  CC=       .3E-11
R    7    4    1  RL
VC   8    4    1  GM=        .5E-1  4
     1    1
     7
```

*********** COMMON EMITTER TRANSISTER AMPLIFIER *************

NUMBER OF NODES= 4
NUMBER OF BRANCHES= 8
NO. OF INPUT TERMINALS= 1
NUMBER OF OUTPUT-TERMINALS = 1
BASE FOR SYMBOL CODES= 4
ELEMENT NO. OF SOURCE = 1

NETWORK

ELEMENT TYPE	ELEMENT NUMBER	INTIAL NODE	TERMINAL NODE	ELEMENT SYMBOL	ELEMENT VALUE	ELEMENT NO. OF CONTROL
I	1	1	2	I1	0.00000E 00	0
R	2	2	1	RS	0.00000E 00	0
R	3	2	3	R1=	0.10000E 03	0
R	4	3	1	R2=	0.10000E 04	0
R	5	3	4	R3=	0.40000E 07	0
C	6	3	4	CC=	0.30000E-11	0
R	7	4	1	RL	0.00000E 00	0
VC	8	4	1	GM=	0.50000E-01	4

TREE SELECTED

R	2	2	1	RS	0.00000E 00	0
R	3	2	3	R1=	0.10000E 03	0
R	5	3	4	R3=	0.40000E 07	0

ELEMENT NUMBER ASSOCIATED WITH OUTPUT= 7

SFG

INITIAL NODE	TERMINAL NODE	EXPONENT OF S	BRANCH VALUE	BRANCH VALUE	1 IF SYMBOL SYMBOL IS INVERTED	1 IF SYMBOL IS USED
7	1	0	-0.10000E 01	FB	0	1
1	2	0	0.10000E 01	RS	0	1
3	4	0	-0.10000E-02	R2	1	0
4	3	0	0.10000E 03	R1	0	0
2	4	0	0.10000E-02	R2	1	0
4	2	0	-0.10000E 01	RS	0	1
5	6	1	0.30000E-11	CC	0	0
6	5	0	-0.40000E 07	R3	0	0
5	7	0	-0.10000E 01	RL	1	1
7	5	0	0.40000E 07	R3	0	0
3	7	0	-0.10000E 01	RL	1	1
7	3	0	0.10000E 03	R1	0	0
2	7	0	0.10000E 01	RL	1	1
7	2	0	-0.10000E 01	RS	0	1
4	9	0	0.10000E 04	R2	0	0
9	8	0	0.50000E-01	GM	0	0
8	5	0	0.40000E 07	R3	0	0
8	3	0	0.10000E 03	R1	0	0
8	2	0	-0.10000E 01	RS	0	1

CIRCUITS

NO.	NODE LIST
1	1 2 7 1
2	1 2 4 9 8 5 7 1
3	1 2 4 9 8 3 7 1
4	1 2 4 3 7 1
5	2 7 3 4 9 8 2
6	2 7 3 4 2
7	2 7 2
8	2 4 9 8 5 7 2
9	2 4 9 8 3 7 2
10	2 4 9 8 2
11	2 4 3 7 2
12	2 4 2
13	3 7 3
14	3 4 9 8 5 7 3
15	3 4 9 8 3
16	3 4 3
17	5 7 5
18	5 6 5

****

NUMERATOR POLYNOMIAL

=(-0.199999E 06+0.12000E-04 S)RS/RL

COLUMN	SYMBOL FOR GIVEN COLUMN
1	RS / RL

POWER OF S	CONSTANT COEFS. IN THE POLYNOMIAL COLUMN 1	COLUMN
0	-0.19999E 06	
1	0.12000E-04	

****

DENOMINATOR POLYNOMIAL

=(0.40010E 04+0.12000E-04 S) RS/RL+(0.50999E-01+0.61199E-06S)RS
+0.44001E 07+0.12000E-02S)1/RL+0.60999E 01+0.73199E-04S

COLUMN	SYMBOL FOR GIVEN COLUMN
1	RS / RL
2	RS / 1
3	1 / RL
4	1 / 1

CONSTANT COEFS. IN THE POLYNOMIAL

POWER OF S	COLUMN 1	COLUMN 2	COLUMN 3	COLUMN 4	COLUMN
0	0.40010E 04	0.50999E-01	0.44001E 07	0.60999E 01	
1	0.12000E-04	0.61199E-06	0.12000E-02	0.73199E-04	

****

APPENDIX C

Worked out examples and listing

************* COMMON EMITTER TRANSISTER AMPLIFIER *************

NUMBER OF NODES= 4
NUMBER OF BRANCHES= 9
NO. OF INPUT TERMINALS= 2
NUMBER OF OUTPUT-TERMINALS = 2
BASE FOR SYMBOL CODES= 4
ELEMENT NO. OF SOURCE = 1
ELEMENT NO. OF SOURCE (1)= 9

NETWORK

ELEMENT TYPE	ELEMENT NUMBER	INTIAL NODE	TERMINAL NODE	ELEMENT SYMBOL	ELEMENT VALUE	ELEMENT NO. OF CONTROL
I	1	1	2	I1	0.00000E 00	0
R	2	2	1	RS	0.00000E 00	0
R	3	2	3	R1=	0.10000E 03	0
R	4	3	1	R2=	0.10000E 04	0
R	5	3	4	R3=	0.40000E 07	0
C	6	3	4	CC=	0.30000E-11	0
R	7	4	1	RL	0.00000E 00	0
VC	8	4	1	GM=	0.50000E-01	4
CC	9	1	4	K1	0.00000E 00	1

TREE SELECTED

R	2	2	1	RS	0.00000E 00	0
R	3	2	3	R1=	0.10000E 03	0
R	5	3	4	R3=	0.40000E 07	0

ELEMENT NUMBER ASSOCIATED WITH OUTPUT(1)= 2

ELEMENT NUMBER ASSOCIATED WITH OUTPUT(2)= 7

MODIFIED NETWORK

ELEMENT TYPE	ELEMENT NUMBER	INTIAL NODE	TERMINAL NODE	ELEMENT SYMBOL	ELEMENT VALUE	ELEMENT NO. OF CONTROL
I	1	1	2	I1	0.00000E 00	0
R	2	2	1	RS	0.00000E 00	0
R	3	2	3	R1=	0.10000E 03	0
R	4	3	1	R2=	0.10000E 04	0
R	5	3	4	R3=	0.40000E 07	0
C	6	3	4	CC=	0.30000E-11	0
R	7	4	1	RL	0.00000E 00	0
VC	8	4	1	GM=	0.50000E-01	4
CC	9	1	4	K1	0.00000E 00	1
VV	10	4	5	AAA	0.00000E 00	2
VV	11	5	6	BBB	0.00000E 00	7

TREE SELECTED

VV	10	4	5	AAA	0.00000E 00	2
VV	11	5	6	BBB	0.00000E 00	7
R	2	2	1	RS	0.00000E 00	0
R	3	2	3	R1=	0.10000E 03	0
R	5	3	4	R3=	0.40000E 07	0

SFG

INITIAL NODE	TERMINAL NODE	EXPONENT OF S	BRANCH VALUE	BRANCH VALUE SYMBOL	1 IF SYMBOL IS INVERTED	1 IF SYMBOL IS USED
14	1	0	-0.10000E 01	FB	0	1
1	2	0	0.10000E 01	RS	0	1
3	4	0	-0.10000E-02	R2	1	0
4	3	0	0.10000E 03	R1	0	0
2	4	0	0.10000E-02	R2	1	0
4	2	0	-0.10000E 01	RS	0	1
5	6	1	0.30000E-11	CC	0	0
6	5	0	-0.40000E 07	R3	0	0
5	7	0	-0.10000E 01	RL	1	1
7	5	0	0.40000E 07	R3	0	0
3	7	0	-0.10000E 01	RL	1	1
7	3	0	0.10000E 03	R1	0	0
2	7	0	0.10000E 01	RL	1	1
7	2	0	-0.10000E 01	RS	0	1
4	12	0	0.10000E 04	R2	0	0
12	8	0	0.50000E-01	GM	0	0
8	5	0	0.40000E 07	R3	0	0
8	3	0	0.10000E 03	R1	0	0
8	2	0	-0.10000E 01	RS	0	1
1	9	0	0.10000E 01	K1	0	1
9	2	0	0.10000E 01	RS	0	1
9	3	0	-0.10000E 03	R1	0	0
9	5	0	-0.40000E 07	R3	0	0
2	10	0	0.10000E 01	AAA	0	1
7	13	0	0.10000E 01	RL	0	1
13	11	0	0.10000E 01	BBB	0	1
10	14	0	0.10000E 01	1	0	0
11	14	0	0.10000E 01	1	0	0

CIRCUITS

NO.	NODE LIST											
1	1	9	5	7	13	11	14	1				
2	1	9	5	7	3	4	12	8	2	10	14	1
3	1	9	5	7	3	4	2	10	14	1		
4	1	9	5	7	2	10	14	1				
5	1	9	3	7	13	11	14	1				
6	1	9	3	7	2	10	14	1				
7	1	9	3	4	12	8	5	7	13	11	14	1
8	1	9	3	4	12	8	5	7	2	10	14	1
9	1	9	3	4	12	8	2	10	14	1		
10	1	9	3	4	12	8	2	7	13	11	14	1
11	1	9	3	4	2	10	14	1				
12	1	9	3	4	2	7	13	11	14	1		
13	1	9	2	10	14	1						
14	1	9	2	7	13	11	14	1				
15	1	9	2	4	12	8	5	7	13	11	14	1
16	1	9	2	4	12	8	3	7	13	11	14	1
17	1	9	2	4	3	7	13	11	14	1		
18	1	2	10	14	1							
19	1	2	7	13	11	14	1					
20	1	2	4	12	8	5	7	13	11	14	1	
21	1	2	4	12	8	3	7	13	11	14	1	
22	1	2	4	3	7	13	11	14	1			
23	2	7	3	4	12	8	2					
24	2	7	3	4	2							
25	2	7	2									
26	2	4	12	8	5	7	2					
27	2	4	12	8	3	7	2					
28	2	4	12	8	2							
29	2	4	3	7	2							
30	2	4	2									
31	3	7	3									
32	3	4	12	8	5	7	3					
33	3	4	12	8	3							
34	3	4	3									
35	5	7	5									
36	5	6	5									

****

NUMERATOR POLYNOMIAL

COLUMN	SYMBOL FOR GIVEN COLUMN
1	K1 RL BBB / RL
2	K1 RS AAA / RL
3	K1 RS AAA / 1
4	K1 RS RL BBB / RL
5	RS AAA / 1
6	RS RL BBB / RL
7	RS AAA / RL

POWER OF S — CONSTANT COEFS. IN THE POLYNOMIAL

POWER OF S	COLUMN 1	COLUMN 2	COLUMN 3	COLUMN 4	COLUMN
0	0.44001E 07	-0.12500E 00	0.10000E 01	0.40010E 04	
1	0.12000E-02	0.00000E 00	0.12000E-04	0.12000E-04	
	COLUMN 5	COLUMN 6	COLUMN 7	COLUMN	
0	0.60999E 01	-0.19999E 06	0.44001E 07		
1	0.73199E-04	0.12000E-04	0.12000E-02		

****

DENOMINATOR POLYNOMIAL

=(0.40010E04+0.1200E-04S)RS/RL+(0.50999E-01+0.61199E-06S)RS
+(0.44001E07+0.12000E-02S)1/RL+0.60999E01+0.73199E-04S

COLUMN	SYMBOL FOR GIVEN COLUMN
1	RS / RL
2	RS / 1
3	1 / RL
4	1 / 1

POWER OF S — CONSTANT COEFS. IN THE POLYNOMIAL

POWER OF S	COLUMN 1	COLUMN 2	COLUMN 3	COLUMN 4	COLUMN
0	0.40010E 04	0.50999E-01	0.44001E 07	0.60999E 01	
1	0.12000E-04	0.61199E-06	0.12000E-02	0.73199E-04	

****

NUMERATOR POLYNOMIAL FOR THE NETWORK FUNCTIONS $V_2/I_1, V_7/I_1, V_2/I_9$ and V_7/I_9 are respectively,

(0.60999E01+0.73199E-04S)RS+(0.44001E-07+0.12000E-02S)RS/RL,
(-0.19999E06+0.12000E-04S)RS,
-0.12500E00RS/RL+(0.10000E-01+0.12000E-04S)RS
and 0.44001E07+0.12000E-02S+(0.40010E04+0.12000E-04S)RS

DINOMINATOR POLYNOMIAL IS SAME AS ABOVE

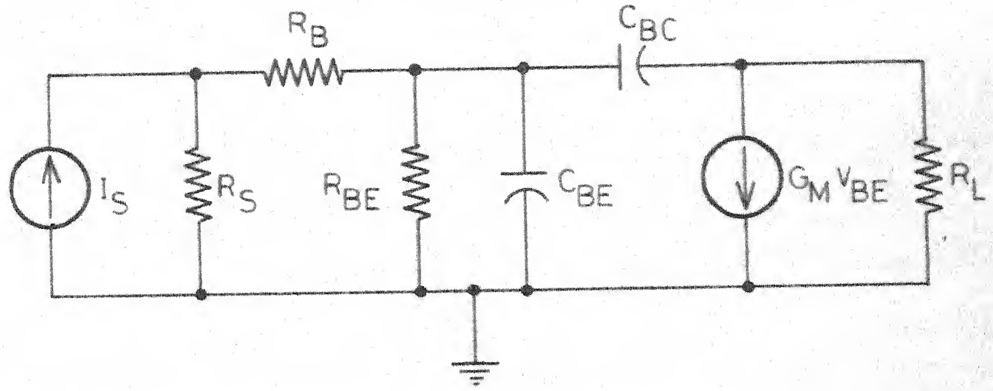

Figure VI.1 Common emitter transistor amplifier.

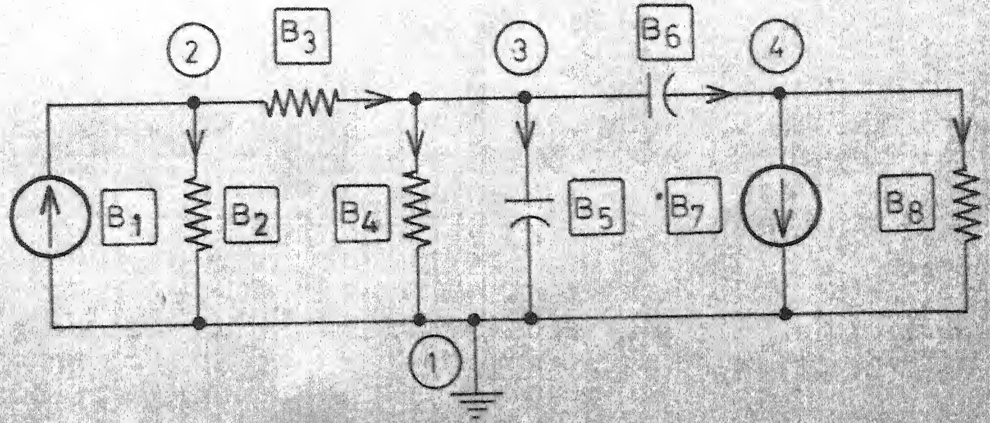

Figure VI.2 Circuit of figure VI.1 with the branches and nodes numbered.

************* COMMON EMITTER TRANSISTOR AMPLIFIER ************

NUMBER OF NODES= 4
NUMBER OF BRANCHES= 8
NO. OF INPUT TERMINALS= 1
NUMBER OF OUTPUT-TERMINALS = 1
BASE FOR SYMBOL CODES= 4
ELEMENT NO. OF SOURCE = 1

NETWORK

ELEMENT TYPE	ELEMENT NUMBER	INTIAL NODE	TERMINAL NODE	ELEMENT SYMBOL	ELEMENT VALUE	ELEMENT NO. OF CONTROL
I	1	1	2	IS	0.00000E 00	0
R	2	2	1	RS=	0.10000E 04	0
R	3	2	3	RB=	0.70000E 02	0
R	4	3	1	RBE	0.00000E 00	0
C	5	3	1	CBE	0.00000E 00	0
C	6	3	4	CBC	0.00000E 00	0
VC	7	4	1	GM	0.00000E 00	4
R	8	1	4	RL=	0.10000E 05	0
TREE SELECTED						
R	2	2	1	RS=	0.10000E 04	0
R	3	2	3	RB=	0.70000E 02	0
C	6	3	4	CBC	0.00000E 00	0

ELEMENT NUMBER ASSOCIATED WITH OUTPUT= 8

SFG

INITIAL NODE	TERMINAL NODE	EXPONENT OF S	BRANCH VALUE	BRANCH VALUE	1 IF SYMBOL SYMBOL IS INVERTED	1 IF SYMBOL IS USED
8	1	0	-0.10000E 01	FB	0	1
1	2	0	0.10000E 04	RS	0	0
3	4	0	-0.10000E 01	RBE	1	1
4	3	0	0.70000E 02	RB	0	0
2	4	0	0.10000E 01	RBE	1	1
4	2	0	-0.10000E 04	RS	0	0
3	5	1	-0.10000E 01	CBE	0	1
5	3	0	0.70000E 02	RB	0	0
2	5	1	0.10000E 01	CBE	0	1
5	2	0	-0.10000E 04	RS	0	0
4	9	0	0.10000E 01	RBE	0	1
9	7	0	0.10000E 01	GM	0	1
7	6	-1	0.10000E 01	CBC	1	1
7	3	0	0.70000E 02	RB	0	0
7	2	0	-0.10000E 04	RS	0	0
2	8	0	-0.10000E-03	RL	1	0
8	2	0	0.10000E 04	RS	0	0
3	8	0	0.10000E-03	RL	1	0
8	3	0	-0.70000E 02	RB	0	0
6	8	0	0.10000E-03	RL	1	0
8	6	-1	-0.10000E 01	CBC	1	1

NO.	NODE LIST									
1	1	2	8	1						
2	1	2	5	3	8	1				
3	1	2	5	3	4	9	7	6	8	1
4	1	2	4	9	7	6	8	1		
5	1	2	4	9	7	3	8	1		
6	1	2	4	3	8	1				
7	2	8	3	5	2					
8	2	8	3	4	9	7	2			
9	2	8	3	4	2					
10	2	8	2							
11	2	5	3	8	2					
12	2	5	3	4	9	7	6	8	2	
13	2	5	3	4	9	7	2			
14	2	5	3	4	2					
15	2	5	2							
16	2	4	9	7	6	8	3	5	2	
17	2	4	9	7	6	8	2			
18	2	4	9	7	3	8	2			
19	2	4	9	7	3	5	2			
20	2	4	9	7	2					
21	2	4	3	8	2					
22	2	4	3	5	2					
23	2	4	2							
24	3	8	3							
25	3	5	3							
26	3	4	9	7	6	8	3			
27	3	4	9	7	3					
28	3	4	3							
29	6	8	6							

****

NUMERATOR POLYNOMIAL

= -0.10000E 00S+0.95367E-06S²xCBE-0.95367E-06SxCBExGM/CBC
+0.10000E 00xGM/CBC+0.95367E-06SxGM+0.95367E-06Sx1/RBE

COLUMN	SYMBOL FOR GIVEN COLUMN
1	1 / 1
2	CBE / 1
3	CBE RBE GM / RBE CBC
4	RBE GM / RBE CBC
5	RBE GM / RBE
6	1 / RBE

POWER OF S — CONSTANT COEFS. IN THE POLYNOMIAL

POWER OF S	COLUMN 1	COLUMN 2	COLUMN 3	COLUMN 4	COLUMN
1	-0.10000E 00	0.00000E 00	-0.95367E-06	0.00000E 00	
2	0.00000E 00	0.95367E-06	0.00000E 00	0.00000E 00	
0	0.00000E 00	0.00000E 00	0.00000E 00	0.10000E 00	

POWER OF S	COLUMN 5	COLUMN 6	COLUMN
1	0.95367E-06	0.95367E-06	
2	0.00000E 00	0.00000E 00	
0	0.00000E 00	0.00000E 00	

****

DENOMINATOR POLYNOMIAL

=0.10699E 04S²xCBE+0.10699E 04SxGM+0.106999E 04Sx1/RBE+0.11070E 01S
+0.19073E-05SxCBExGM/CBC+0.10000E-03x1/CBC+0.10700E 00xCBE/CBC+0.10700E 00x
(1/RBExCBC)

```
COLUMN                 SYMBOL FOR GIVEN COLUMN
   1                          CBE    /  1
   2                          RBE     GM     / RBE
   3                           1     / RBE
   4                           1     /  1
   5                          CBE   RBE     GM     / RBE    CBC
   6                          CBE   RBE     GM     / RBE
   7                          CBE    / RBE
   8                          RBE     GM     / RBE    CBC
   9                           1     / CBC
  10                          CBE    / RBE    CBC
  11                          CBE    / CBC
  12                           1     / RBE    CBC


POWER
 OF S
                          CONSTANT COEFS.IN THE POLYNOMIAL
         COLUMN 1        COLUMN 2        COLUMN 3        COLUMN 4        COLUMN
   1    0.00000E 00     0.10699E 04     0.10699E 04     0.11070E 01
   2    0.10699E 04     0.00000E 00     0.00000E 00     0.00000E 00
   0    0.00000E 00     0.00000E 00     0.00000E 00     0.00000E 00
         COLUMN 5        COLUMN 6        COLUMN 7        COLUMN 8        COLUMN
   1    0.19073E-05     0.00000E 00     0.00000E 00     0.00000E 00
   2    0.00000E 00     0.00000E 00     0.00000E 00     0.00000E 00
   0    0.00000E 00     0.00000E 00     0.00000E 00     0.00000E 00
         COLUMN 9        COLUMN10        COLUMN11        COLUMN12        COLUMN
   1    0.00000E 00     0.00000E 00     0.10700E 00     0.00000E 00
   2    0.00000E 00     0.00000E 00     0.00000E 00     0.00000E 00
   0    0.10000E-03     0.00000E 00     0.00000E 00     0.10700E 00
*****************************************************


NUMBER OF SETS = 4
NUMBER OF FREQUENCYS = 9


*****************************************************
```

SET NUMBER = 1

SYMBOL (1) = 1
SYMBOL (1) = 0.10000E 01
SYMBOL (2) =CBE
SYMBOL (2) = 0.10000E-08
SYMBOL (3) =RBE
SYMBOL (3) = 0.15000E 04
SYMBOL (4) = GM
SYMBOL (4) = 0.35000E-01
SYMBOL (5) =CBC
SYMBOL (5) = 0.15000E-10

REAL VALUE OF NUMERATOR = 0.23333E 09
IMAGINARY VALUE OF NUMERATOR =-0.10000E 00
REAL VALUE OF DENOMINATOR = 0.11422E 08
IMAGINARY VALUE OF DENOMINATOR = 0.46403E 02

FREQUENCY	AMPLITUDE	PHASE ANGLE
0.10000E 01	0.20428E 02	-0.25528E-04
0.10000E 02	0.20428E 02	-0.25528E-03
0.10000E 03	0.20427E 02	-0.25528E-02
0.10000E 04	0.20421E 02	-0.25523E-01
0.10000E 05	0.19793E 02	-0.24994E 00
0.10000E 06	0.74514E 01	-0.11976E 01
0.10000E 07	0.79967E 00	-0.15343E 01
0.10000E 08	0.80057E-01	0.15477E 01
0.10000E 09	0.82879E-02	0.13081E 01
0.10000E 10	0.22988E-02	0.35560E 00

****

SET NUMBER = 2

SYMBOL (1) = 0.10000E 01
SYMBOL (2) = 0.50000E-09
SYMBOL (3) = 0.30000E 04
SYMBOL (4) = 0.17500E-01
SYMBOL (5) = 0.15000E-10

REAL VALUE OF NUMERATOR = 0.11666E 09
IMAGINARY VALUE OF NUMERATOR =-0.10000E 00
REAL VALUE OF DENOMINATOR = 0.90444E 07
IMAGINARY VALUE OF DENOMINATOR = 0.23755E 02

FREQUENCY	AMPLITUDE	PHASE ANGLE
0.10000E 01	0.12899E 02	-0.16508E-04
0.10000E 02	0.12899E 02	-0.16508E-03
0.10000E 03	0.12899E 02	-0.16508E-02
0.10000E 04	0.12897E 02	-0.16506E-01
0.10000E 05	0.12727E 02	-0.16360E 00
0.10000E 06	0.66848E 01	-0.10265E 01
0.10000E 07	0.78021E 00	-0.15156E 01
0.10000E 08	0.78275E-01	0.15230E 01
0.10000E 09	0.88778E-02	0.10773E 01
0.10000E 10	0.42815E-02	0.18364E 00

****

SET NUMBER = 3

SYMBOL (1) = 0.10000E 01
SYMBOL (2) = 0.20000E-08
SYMBOL (3) = 0.75000E 03
SYMBOL (4) = 0.70000E-01
SYMBOL (5) = 0.10000E-10

REAL VALUE OF NUMERATOR = 0.69999E 09
IMAGINARY VALUE OF NUMERATOR =-0.10001E 00
REAL VALUE OF DENOMINATOR = 0.24266E 08
IMAGINARY VALUE OF DENOMINATOR = 0.98833E 02

FREQUENCY	AMPLITUDE	PHASE ANGLE
0.10000E 01	0.28846E 02	-0.25591E-04
0.10000E 02	0.28846E 02	-0.25591E-03
0.10000E 03	0.28846E 02	-0.25591E-02
0.10000E 04	0.28836E 02	-0.25585E-01
0.10000E 05	0.27945E 02	-0.25053E 00
0.10000E 06	0.10499E 02	-0.11983E 01
0.10000E 07	0.11263E 01	-0.15326E 01
0.10000E 08	0.11272E 00	0.15657E 01
0.10000E 09	0.11317E-01	0.14816E 01
0.10000E 10	0.15148E-02	0.83928E 00

****

SET NUMBER = 4

SYMBOL (1) = 0.10000E 01
SYMBOL (2) = 0.10000E-09
SYMBOL (3) = 0.15000E 05
SYMBOL (4) = 0.50000E-02
SYMBOL (5) = 0.27000E-10

REAL VALUE OF NUMERATOR = 0.18518E 08
IMAGINARY VALUE OF NUMERATOR =-0.99999E-01
REAL VALUE OF DENOMINATOR = 0.39679E 07
IMAGINARY VALUE OF DENOMINATOR = 0.69246E 01

FREQUENCY	AMPLITUDE	PHASE ANGLE
0.10000E 01	0.46670E 01	-0.10999E-04
0.10000E 02	0.46670E 01	-0.10999E-03
0.10000E 03	0.46670E 01	-0.10999E-02
0.10000E 04	0.46668E 01	-0.10998E-01
0.10000E 05	0.46392E 01	-0.10955E 00
0.10000E 06	0.31448E 01	-0.83479E 00
0.10000E 07	0.42411E 00	-0.15137E 01
0.10000E 08	0.44944E-01	0.12528E 01
0.10000E 09	0.15055E-01	0.28752E 00
0.10000E 10	0.14447E-01	0.29555E-01

```
***************************************************
MAX. VALUE OF AMPLITUDE = 0.28846E 02
MIN. VALUE OF AMPLITUDE = 0.15148E-02
MAX. VALUE OF PHASE-ANGLE = 0.15657E 01
MIN. VALUE OF PHASE-ANGLE =-0.15343E 01
***************************************************
```

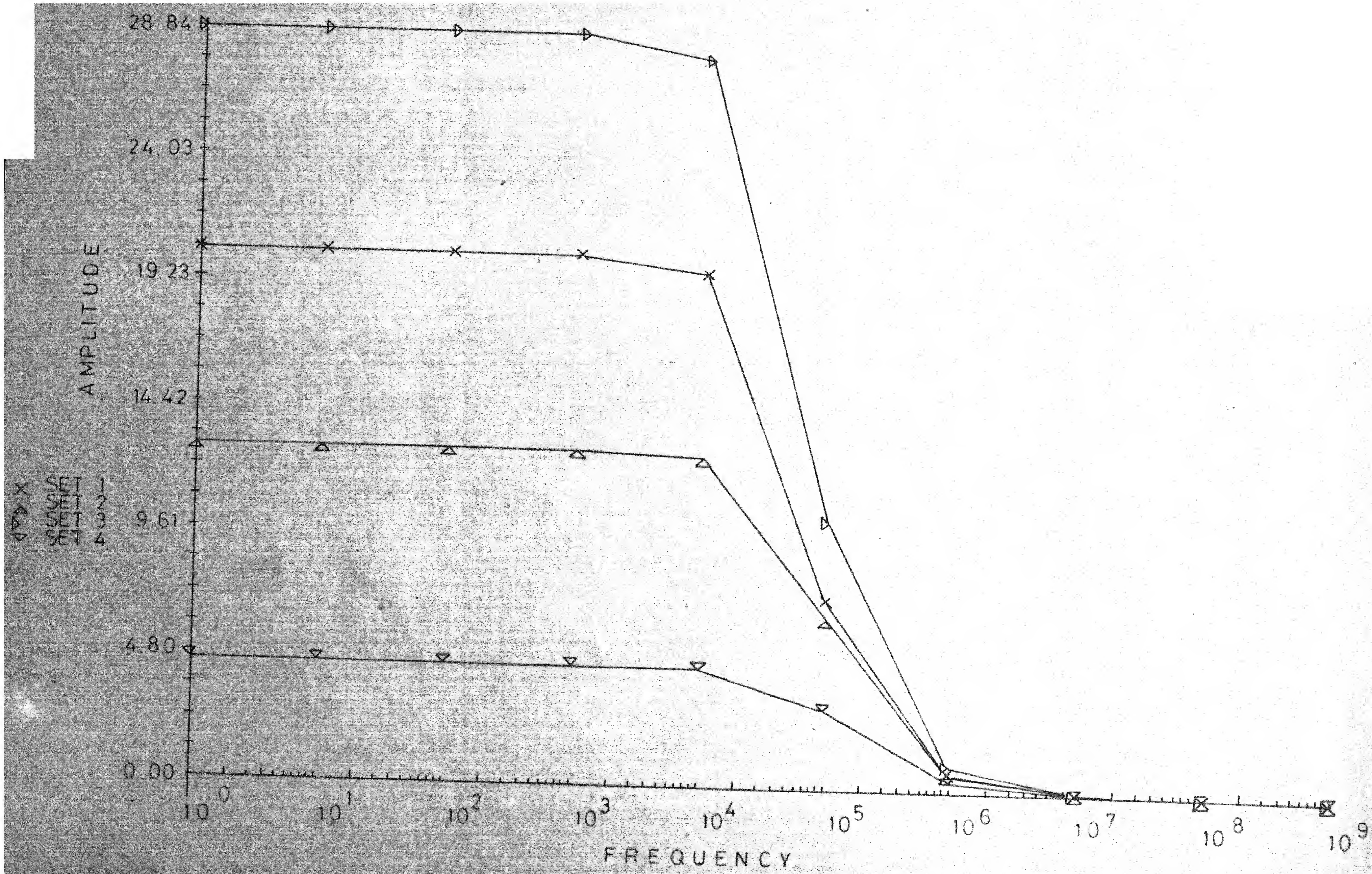

Figure VI.3 Amplitude of frequency response of common emitter transistor stage.

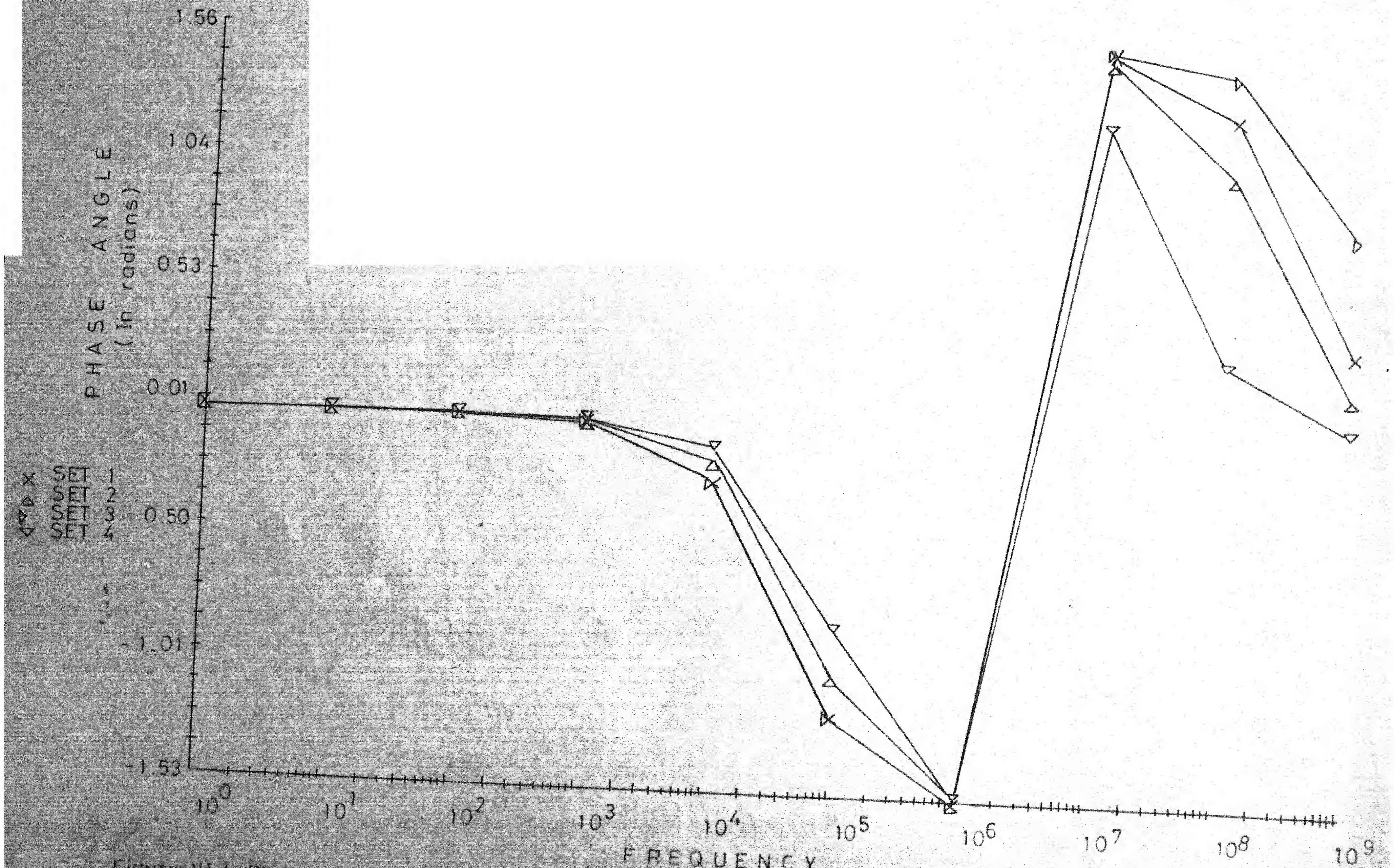

Figure VI.4 Phase angle of frequency response of common emitter transistor stage

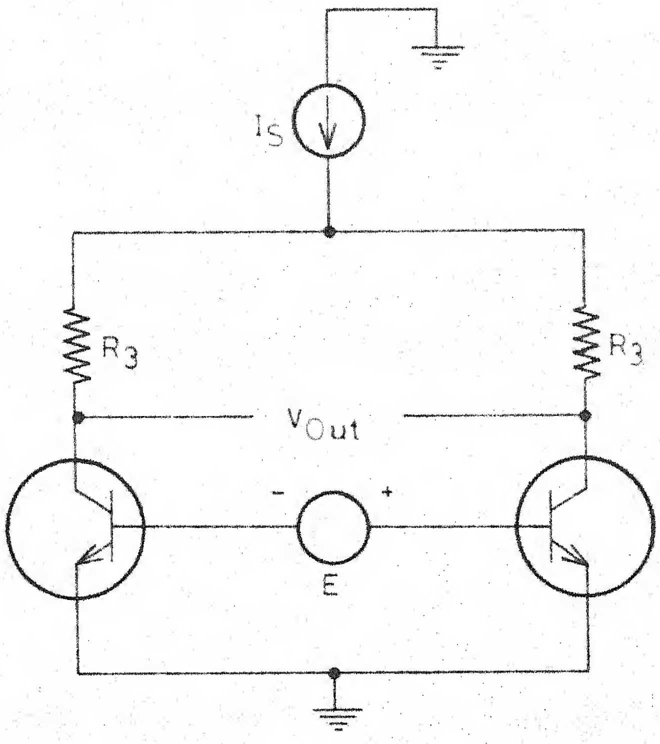

Figure VI.5 Differential amplifier.

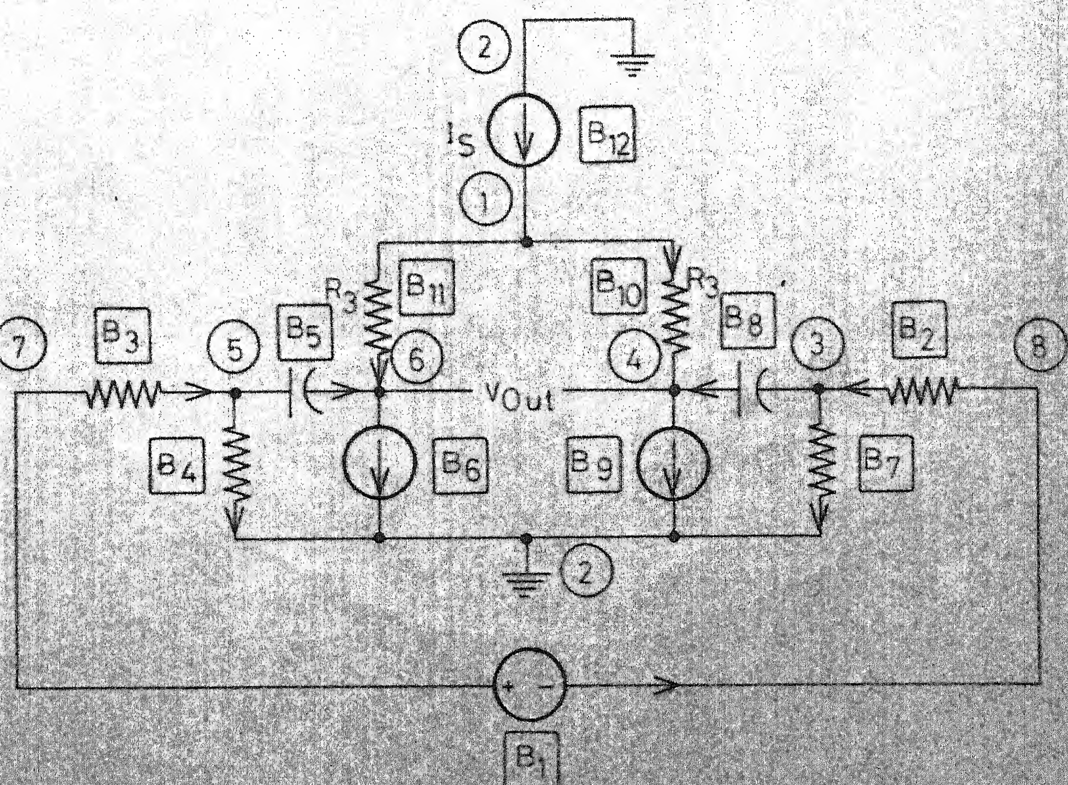

Figure VI.6 Equivalent circuit of the figure VI.5.

SFG

INITIAL NODE	TERMINAL NODE	EXPONENT OF S	BRANCH VALUE	,BRANCH SYMBOL	1 IF SYMBOL IS INVERTED	1 IF SYMBOL IS USED
13	1	0	-C.10000E 01	FB	0	1
4	6	0	0.10000E 01	GM	0	1
6	5	-1	C.10000E 01	C2	1	1
6	4	0	-C.10000E 01	R2	0	1
2	7	0	-C.10000E 01	R2	1	1
7	2	0	0.17000E 03	R1	0	0
1	7	0	-C.10000E 01	R2	1	1
3	7	0	C.10000E 01	R2	1	1
7	3	0	-0.17000E 03	R1	0	0
4	7	0	0.10000E 01	R2	1	1
7	4	0	-C.10000E 01	R2	0	1
8	9	0	0.10000E 01	GM	0	1
9	8	-1	0.10000E 01	C2	1	1
9	2	0	C.17000E 03	R1	0	0
9	3	0	-C.17000E 03	R1	0	0
9	4	0	-0.10000E 01	R2	0	1
10	11	0	0.10000E 01	R3	1	1
11	10	0	-C.10000E 01	R3	0	1
8	11	0	-0.10000E 01	R3	1	1
11	8	-1	0.10000E 01	C2	1	1
2	11	0	-C.10000E 01	R3	1	1
11	2	0	C.17000E 03	R1	0	0
1	11	0	-0.10000E 01	R3	1	1
3	11	0	C.10000E 01	R3	1	1
11	3	0	-C.17000E 03	R1	0	0
5	11	0	0.10000E 01	R3	1	1
11	5	-1	-0.10000E 01	C2	1	1
12	4	0	0.10000E 01	R2	0	1
12	3	0	0.17000E 03	R1	0	0
12	2	0	-0.17000E 03	R1	0	0
12	8	-1	-C.10000E 01	C2	1	1
12	10	0	C.10000E 01	R3	0	1
5	13	0	-0.10000E 01	1	0	0
3	13	0	-0.10000E 01	1	0	0
1	13	0	C.10000E 01	1	0	0
2	13	0	C.10000E 01	1	0	0
8	13	0	0.10000E 01	1	0	0

NO.	NODE LIST										
1	1	13	1								
2	1	11	8	13	1						
3	1	11	8	9	4	7	3	13	1		
4	1	11	8	9	4	7	2	13	1		
5	1	11	8	9	4	6	5	13	1		
6	1	11	8	9	3	13	1				
7	1	11	8	9	3	7	4	6	5	13	1
8	1	11	8	9	3	7	2	13	1		
9	1	11	8	9	2	13	1				
10	1	11	8	9	2	7	4	6	5	13	1
11	1	11	8	9	2	7	3	13	1		
12	1	11	5	13	1						
13	1	11	3	13	1						
14	1	11	3	7	4	6	5	13	1		
15	1	11	3	7	2	13	1				
16	1	11	2	13	1						
17	1	11	2	7	4	6	5	13	1		
18	1	11	2	7	3	13	1				
19	1	7	4	6	5	13	1				
20	1	7	4	6	5	11	8	13	1		
21	1	7	4	6	5	11	8	9	3	13	1
22	1	7	4	6	5	11	8	9	2	13	1
23	1	7	4	6	5	11	3	13	1		
24	1	7	4	6	5	11	2	13	1		
25	1	7	3	13	1						
26	1	7	3	11	8	13	1				
27	1	7	3	11	8	9	4	6	5	13	1
28	1	7	3	11	8	9	2	13	1		
29	1	7	3	11	5	13	1				
30	1	7	3	11	2	13	1				
31	1	7	2	13	1						
32	1	7	2	11	8	13	1				
33	1	7	2	11	8	9	4	6	5	13	1
34	1	7	2	11	8	9	3	13	1		
35	1	7	2	11	5	13	1				
36	1	7	2	11	3	13	1				
37	2	11	8	9	4	7	2				
38	2	11	8	9	3	7	2				
39	2	11	8	9	2						
40	2	11	3	7	2						
41	2	11	2								
42	2	7	4	6	5	11	8	9	2		
43	2	7	4	6	5	11	2				
44	2	7	3	11	8	9	2				
45	2	7	3	11	2						
46	2	7	2								
47	3	11	8	9	4	7	3				
48	3	11	8	9	3						
49	3	11	3								
50	3	7	4	6	5	11	8	9	3		
51	3	7	4	6	5	11	3				
52	3	7	3								
53	4	7	4								
54	4	6	5	11	8	9	4				
55	4	6	4								
56	5	11	5								
57	8	11	8								
58	8	9	8								
59	10	11	10								

****

NUMERATOR POLYNOMIAL

COLUMN	SYMBOL FOR GIVEN COLUMN
1	1 / 1
2	1 / R3 C2
3	GM R2 / R3 R2 C2
4	GM**2 R2 / R3 C2**2
5	GM / R3 C2
6	GM**2 R2 / R3 R2 C2**2
7	GM / R3 R2 C2
8	1 / R3
9	1 / R3 R2
10	GM R2 / R2 C2
11	GM R2 / R3 R2 C2**2
12	1 / R2
13	1 / R3 R2 C2
14	R2 / R2
15	GM R2 / 1
16	GM / C2
17	R3 / R3
18	R2 / R3 R2 C2
19	GM R2 / R3 C2
20	GM**2 R2 / R3 C2
21	GM**2 R2 / R3 R2 C2
22	GM / R3 C2**2
23	R2 / R3 R2
24	GM R2 / R3
25	GM R2 / R3 R2
26	GM**2 R2 / R2 C2**2
27	GM R2 R3 / R3 R2 C2
28	GM R2 / R2
29	GM / R2 C2
30	R3 / R3 R2
31	GM / R3 R2 C2**2
32	GM**2 R2 / R2 C2
33	GM R2 R3 / R3 R2
34	GM**2 R2 R3 / R3 R2 C2
35	GM R3 / R3 R2 C2
36	R2 R3 / R3 R2
37	GM**2 R2 / C2
38	GM R2 R3 / R3
39	GM**2 R2 R3 / R3 C2
40	GM R3 / R3 C2
41	GM**2 R2 R3 / R3 R2 C2**2

```
POWER
OF S                   CONSTANT COEFS, IN THE POLYNOMIAL
          COLUMN 1          CCLUMN 2          CCLLMN 3          COLUMN 4          COLUMN
   2      0.10000E 01       0.                0.                0.
   1      0.                0.               -0.                0.
   0      0.                0.                0.                0.
 NEGATIVE OUTPUT VOLTAGE TERMINAL=  4
          COLUMN 5          COLUMN 6          CCLLMN 7          COLUMN 8          COLUMN
   2      0.                0.                0.                C.
   1     -0.                0.                0.                0.
   0      0.               -0.                0.                0.
 NEGATIVE OUTPUT VOLTAGE TERMINAL=  4
          COLUMN 9          CCLUMN10          CCLLMN11          COLUMN12          COLUMN
   2      0.                0.                0.                0.
   1      0.               -0.20000E 01       0.                0.
   0      0.                0.                0.                0.
 NEGATIVE OUTPUT VOLTAGE TERMINAL=  4
          COLUMN13          COLUMN14          CCLUMN15          COLUMN16          COLUMN
   2      0.                0.10000E 01       0.1C000E 01       0.
   1      0.                C.                0.               -0.10000E 01
   0      0.                0.                0.                0.
 NEGATIVE OUTPUT VOLTAGE TERMINAL=  4
          COLUMN17          COLUMN18          CCLLMN19          COLUMN20          COLUMN
   2      0.10000E 01       0.                0.                0.
   1      0.                0.                0.                0.
   0      0.                0.                0.                0.
 NEGATIVE OUTPUT VOLTAGE TERMINAL=  4
          COLUMN21          COLUMN22          COLUMN23          COLUMN24          COLUMN
   2      0.                0.                0.                0.
   1     -0.                0.                0.                0.
   0      0.               -0.                0.                0.
 NEGATIVE OUTPUT VOLTAGE TERMINAL=  4
          COLUMN25          COLUMN26          CCLLMN27          COLUMN28          COLUMN
   2     -0.                0.                0.                0.
   1      0.                0.               -0.2C000E 01       0.
   0      0.                0.10000E 01       0.                0.
 NEGATIVE OUTPUT VOLTAGE TERMINAL=  4
          COLUMN29          COLUMN30          COLUMN31          COLUMN32          COLUMN
   2      0.                0.                0.                0.
   1     -0.                0.                0.                0.
   0      0.                0.                0.                0.
 NEGATIVE OUTPUT VOLTAGE TERMINAL=  4
          COLUMN33          COLUMN34          CCLUMN35          COLUMN36          COLUMN
   2     -0.                0.                0.                0.10000E 01
   1      0.                0.                0.                0.
   0      0.                0.                0.                0.
 NEGATIVE OUTPUT VOLTAGE TERMINAL=  4
          COLUMN37          COLUMN38          COLUMN39          COLUMN40          COLUMN
   2      0.                0.10000E 01       0.                0.
   1     -0.10000E 01       0.               -0.1C000E 01      -0.10000E 01
   0      0.                0.                0.                0.
 NEGATIVE OUTPUT VOLTAGE TERMINAL=  4
          COLUMN41          CCLUMN
   2      0.
   1      0.
   0      0:10000E 01
*****************************************************
```

DENOMINATOR POLYNOMIAL

COLUMN	SYMBOL FOR GIVEN COLUMN
1	GM R2 / R3 R2 C2
2	GM / R3 R2 C2
3	GM / R3 C2
4	1 / R3 R2
5	1 / R3
6	GM**2 R2 / R3 R2 C2**2
7	1 / R2
8	R2 / R2
9	GM**2 R2 / R3 C2**2
10	GM R2 / 1
11	1 / R3 C2
12	GM / C2
13	R3 / R3
14	GM**2 R2 / R3 R2 C2
15	GM**2 R2 / R3 C2
16	GM R2 / R3 R2
17	R2 / R3 R2
18	GM R2 / R3
19	GM R2 / R2
20	1 / R3 R2 C2
21	GM / R2 C2
22	R3 / R3 R2
23	R2 / R3 R2 C2
24	GM R2 / R2 C2
25	R2 R3 / R3 R2
26	GM R2 / R3 C2
27	GM**2 R2 / C2
28	GM R2 R3 / R3
29	GM / R3 C2**2
30	GM R3 / R3 C2
31	GM**2 R2 / R2 C2
32	GM R2 R3 / R3 R2
33	GM**2 R2 R3 / R3 R2 C2
34	GM / R3 R2 C2**2
35	GM R3 / R3 R2 C2
36	GM R2 / R3 R2 C2**2
37	GM R2 R3 / R3 R2 C2
38	GM**2 R2 R3 / R3 C2
39	1 / 1

CONSTANT COEFS.IN THE POLYNOMIAL

POWER OF S					
	COLUMN 1	COLUMN 2	COLUMN 3	COLUMN 4	COLUMN
2	0.	0.	0.	0.	
1	0.	-0.	-0.	0.	
0	0.	0.	0.	0.	
	COLUMN 5	COLUMN 6	COLUMN 7	COLUMN 8	COLUMN
2	0.34000E 03	0.	0.34000E 03	0.10000E 01	
1	0.	0.	0.	0.	
0	0.	-0.	0.	0.	
	COLUMN 9	COLUMN10	COLUMN11	COLUMN12	COLUMN
2	0.	0.10000E 01	0.	0.	
1	0.	0.	0.20000E 01	-0.10000E 01	
0	-0.	0.	0.	0.	
	COLUMN13	COLUMN14	COLUMN15	COLUMN16	COLUMN
2	0.10000E 01	0.	0.	0.	
1	0.	-0.	-0.	0.	
0	0.	0.	0.	0.	
	COLUMN17	COLUMN18	COLUMN19	COLUMN20	COLUMN
2	0.34000E 03	0.34000E 03	0.34000E 03	0.	
1	0.	0.	0.	0.68000E 03	
0	0.	0.	0.	0.	
	COLUMN21	COLUMN22	COLUMN23	COLUMN24	COLUMN
2	0.	0.34000E 03	0.	0.	
1	-0.34000E 03	0.	0.20000E 01	-0.10000E 01	
0	0.	0.	0.	0.	
	COLUMN25	COLUMN26	COLUMN27	COLUMN28	COLUMN
2	0.10000E 01	0.	0.	0.10000E 01	
1	0.	0.20000E 01	-0.10000E 01	0.	
0	0.	0.	0.	0.	
	COLUMN29	COLUMN30	COLUMN31	COLUMN32	COLUMN
2	0.	0.	0.	0.34000E 03	
1	0.	-0.10000E 01	-0.34000E 03	0.	
0	-0.10000E 01	0.	0.	0.	
	COLUMN33	COLUMN34	COLUMN35	COLUMN36	COLUMN
2	0.	0.	0.	0.	
1	-0.34000E 03	0.	-0.34000E 03	0.	
0	0.	-0.34000E 03	0.	-0.10000E 01	
	COLUMN37	COLUMN38	COLUMN39	COLUMN	
2	0.	0.	0.10000E 01		
1	-0.10000E 01	-0.10000E 01	0.		
0	0.	0.	0.		

```
C
C
C
C
C
C
C
C
C
C
C
C
C
C
C
C
C***********************************************************************
C
C
C
C                                 ****SNAP****
C          THIS PROGRAM FINDS THE SYMBOLIC TRANSFER
C          FUNCTION OR IMMITANCE FUNCTION OF A
C          LUMPED LINEAR TIME INVARIANT NETWORK.
C
C
C***********************************************************************
C     THE FOLLOWING ARRAY ARE ASSOCIATED WITH THE NETWORK
C     CHARACTERISTIC NBN(DEFINED IN PRORAM MAIN -1)
       DIMENSION LT(25),IG(25),SMBOL(75)
      DIMENSION IFLOW(25),NP(25),KODES(25),KONC(25)
      DIMENSION N(25,25),CONS(25,25),KODE(25,25),IXPO(25,25)
C***********************************************************************
C     THE FOLLOWING ARRAYS ARE ASSOCIATED WITH THE NETWORK
C     CHARATERISTIC NBG
      DIMENSION NFIRST(75),NLAST(75),IXPON(75),WEIGT(75)
      DIMENSION SYMBUL(75),MIX(75),CVAL(75)
      DIMENSION KONSC(75),NEST(75),TYPE(75)
C***********************************************************************
C     THE FOLLOWING ARRAYS ARE ASSOCIATED WITH THE NETWORK
C      CHARACTERISTIC NPAC
      DIMENSION CONST(220),KODET(220),IXPOT(220),MAPO(220)
      DIMENSION NOCTCT(220),NUP(220),JAC(220)
      DIMENSION NPCODE(220)
C***********************************************************************
C     THE FOLLOWING ARRAYS ARE ASSOCIATED WITH THE NETWORK
C     CHARACTERISTIC NTC ,NSPT, AND NEXPS
      DIMENSION NA(125),NB(125)
      DIMENSION KONS(16),KODI(16),SEMBOL(16),KODF(16)
      DIMENSION MSORT(12),KSORT(125),POLYU(12,125)
```

```
      DIMENSION POLY(12,125),ITOP(125)                                  SNA00510
      DIMENSION SIMBCN(125,8),SIMBCC(125,8)                             SNA00520
      DIMENSIONSEMPCN(125,8),SEMPCC(125,8)                              SNA00530
      DIMENSION ISET(12,75),NCTCH(900),STAR(9)                          SNA00540
C***********************************************************************SNA00550
C     THE FOLLOWING ARRAYS ARE ASSOCIATED WITH THE NETWORK              SNA00560
C     CHARACTERISTICS NRI,NCI,NEON,AND NRS                              SNA00570
C                                                                       SNA00580
       DIMENSION SM(6),SN(6)                                            SNA00541
      COMMON /C1/MSORT,KSORT                                            SNA00590
      COMMON /C2/NNG,NBG                                                SNA00600
      COMMON /C3/NEXPS,NTO                                              SNA00610
      COMMON /C4/NSPT                                                   SNA00620
       COMMON/C5/SM,SN,NCOUT,NINN                                       SNA00622
        EQUIVALENCE (IXPO(1,1),NCTCH(1),SIMBCN(1,1))                    SNA00630
      EQUIVALENCE (CONS(1,1),ISET(1,1),SIMBOC(1,1))                     SNA00640
      EQUIVALENCE (KODE(1,1),POLYU(1,1))                                SNA00650
      REAL IBLANK                                                       SNA00660
      DATA DASH/2H //                                                   SNA00670
      DATA FB,SB/3H FB,3H 1 /                                           SNA00680
      DATA STAR(1),STAR(2),STAR(3)/3H   ,3H**2,3H**3/                   SNA00690
      DATA STAR(4),STAR(5),STAR(6)/3H**4,3H**5,3H**6/                   SNA00700
      DATA STAR(7),STAR(8),STAR(9)/3H**7, 3H**8,3H**9/                  SNA00710
       DATA SM(1),SM(2),SM(3)/3HAAA,3HBBB,3HCCC/                        SNA00711
       DATA SM(4),SM(5),SM(6)/3HDDD,3HEEE,3HFFF/                        SNA00712
       DATA SN(1),SN(2),SN(3)/3H K1,3H K2,3H K3/                        SNA00713
       DATA SN(4),SN(5),SN(6)/3H K4,3H K5,3H K6/                        SNA00714
      DATA ONE/3H  1/                                                   SNA00720
      DATA IBLANK/1H /                                                  SNA00730
C                    PROGRAM ---MAIN                                    SNA00740
C                     PRELIMINARY INPUT INFORMATION                     SNA00750
C     NBN=NUMBER OF BRANCHES IN NETWORK.                                SNA00760
C     NBG=NO. OF BRANCHES  OF SFG                                       SNA00770
C     NTO=NO. OF TERMS IN OUTPUT.                                       SNA00780
C     NSPT=NO . OF SYMBOLS PER TERM.                                    SNA00790
C     NEXPS=NO. OF DIFFERENT POWERS OF S                                SNA00800
C     NPAC=NO. OF PATHS PLUS CIRCUITS.                                  SNA00810
C     NRI=MAXIMUM NUMBER OF NONTOUCHING LOOPS.                          SNA00820
C     NCI=MAXIMUM NUMBER OF LOOPS NOT TOUCHING ANY GIVEN LOOP           SNA00830
C     NEON=NUMBER OF NONTOUCHING PAIRS OF LOOPS                         SNA00840
C     NRS=NUMBER OF REPEATED SYMBOLS (NUMBER OF NETWORK                 SNA00850
C     ELEMENTS ASSIGNED SAME SYMBOL)                                    SNA00860
      NBN=25                                                            SNA00870
      NBG=75                                                            SNA00880
      NTO=125                                                           SNA00890
      NSPT=16                                                           SNA00900
      NEXPS=12                                                          SNA00910
      NPAC=220                                                          SNA00920
      NRI=12                                                            SNA00930
      NCI=75                                                            SNA00940
```

```
      NEON=900
      NRS=9
C     NSPTL=NUMBER OF SYMBOLS IN NUMERATOR OF EACH TERM
C     NBTG=NUMBER OF BRANCHES OF TREE OF SFG
C     NNG=NUMBER OF NODES IN SFG
      NNG=NBN
      NSPTL=NSPT/2
      NBTG=NBN
 1111 CONTINUE
      WRITE(6,519)
  519 FORMAT(1H1)
C     THE NEXT 6 CARDS ARE FOR PROBLEM IDENTIFICATION ON THE 1ST DATA CA
      READ (5,1150)(WEIGT(J),J=1,72)
 1150 FORMAT(72A1)
      IF( WEIGT(1).EQ.IBLANK)STOP
      WRITE (6,1160)(WEIGT(J),J=1,71)
 1160 FORMAT(1X,71A1//)
      DO 1151 J=1,72
 1151 WEIGT(J)=0.
      READ(5,1240)NOD,NOB,KBASIS ,LISTG,LISTC,LISTP
 1240 FORMAT(3I5,5X,3I1)
      IF (KBASIS)1357,1357,1358
 1357 KBASIS =8
 1358 CONTINUE
       READ(5,1)NINN,NOOUT
  1   FORMAT(2I5)
C
      WRITE(6,720)NOD
  720 FORMAT(2X,*NUMBER OF NODES=*,I3)
      WRITE(6,721)NOB
  721 FORMAT(2X,*NUMBER OF BRANCHES=*,I3)
      IF(LISTG)723,723,722
  722  CONTINUE
C     LIST SFG
  723  IF (LISTC)725,725,724
  724 CONTINUE
C     LIST ALL CIRCUITS
  725  IF(LISTP)726,726,727
  727   WRITE(6,728)
  728  FORMAT(2X,*LIST ALL PATHS FROM NODE*,       I3,2X,*TO NODE *,I3)
726    WRITE(6,729)NINN
 729  FORMAT(2X,*NO. OF INPUT TERMINALS=*,I3)
C
C
C
      WRITE(6,730)NOOUT
 730  FORMAT(2X,* NUMBER OF OUTPUT-TERMINALS = *,I3)
C
C
C
```

```
C
C
C
C
C
      WRITE(6,850)KBASIS
  850  FORMAT(2X,*BASE FOR SYMBOL CODES=*,I4)
C                    PROGRAM MAIN -2
C                TAKE SFG BRANCH INFORMATION AS FOUND
C                BY SUBROUTINE AND GENERATE
C                    (1)ROUTING MATRIX INFORMATION
C                    N(J,K),AND LT(J)
C                    (2)SFG BRANCH VALUES IXPO(J,L),CONS(J,L),
C                    KODE(J,L)WHERE J=NFIRST(I),L=NLAST(I),AND
C                    I=BRANCH NUMBER
C                    TOGETHER WITH THE SYMBOL SEMBOL(K),K=1,2,....,M1
C
C     CALLSUBROUTINE TO FORMULATE THE SIGNAL FLOW GRAPH,SFG
        CALL SFG(NFIRST,NLAST,IXPON,WEIGT,SYMBUL,KONSO,MIX,NEST,LIST ,
     1NIN,NOUT,NOD ,NOB,LISTG,NODA,NODB)
      IF(NOB)1111,1111,1920
 1920  CONTINUE
      IBO=0
      KC=0
      MICH=1
      K=0
      MG=1
      JLAS=1
      NCIR=1
      ININ=NIN
      INOUT=NOUT
      KCO=0
      DO 301 INK=1,NSPT
  301  KONS(INK)=0
      DO 300 INK=1,NNG
  300 IG(INK)=0
C     FIND IXPO(J,L),CONS (J,L)
      GO TO 307
  305  MG=KBASIS *MG
      MICH=MICH+1
  307  IBO=IBO+1
      IF(LIST-IBO)19,4,4
    4  CONTINUE
      LOB=MIX(IBO)
      J=NFIRST(LOB)
      L=NLAST(LOB)
      IXPO(J,L)=IXPON(LOB)
      CONS(J,L)=WEIGT(LOB)
C     FIND ROUTING MATRIX
    8  IF(J.EQ.JLAS)GO TO 10
```

```
      LT(JLAS)=K
      K1=K+1
      IF(JLAS-MIN)28,27,28
   27 N(JLAS,K1)=-1
      GO TO 29
   28 N(JLAS,K1)=0
   29 JLAS=JLAS+1
      K=0
      GO TO 8
   10 K=K+1
      N(J,K)=L
C     FIND KODE(J,L) AND SEMBOL(KCO)
      SMBOL(IBC)=SYMBUL(LOB)
      MODE=NEST(LOB)
      IF(MODE)335,316,335
  335 IF(IG(L))5,960,5
    5 KODE(J,L)=IG(L)
      GO TO 307
  960 CONTINUE
      KPU=IBC-1
      IF(KPU)953,953,315
  315  DO 952 KP=1,KPU
      IF(SMBOL(IBC).NE.SMBOL(KP))GO TO 952
      LOBX=MIX(KP)
      IF(KONSO(LOB)-KONSO(LOBX))952,956,952
  956 LX=NLAST(LOBX)
      KODE(J,L)=IG(LX)
      GO TO 307
  952  CONTINUE
      IF(SMBOL(IBC).EQ.ONE)GO TO 316
  953  IG(L)=MG
      KCO=KCO+1
      SEMBOL(KCO)=SMBOL(IBC)
      KODE(J,L)=IG(L)
      IF(KONSO(LOB ))3,3,2
    2 KONS(KCO)=1
    3 CONTINUE
      GO TO 305
  316  KODE(J,L)=0
      GO TO 307
   19 LT(JLAS)=K
      K11=K+1
      N(JLAS,K11)=0
C     C
C                        PROGRAM MAIN--3
C                        NULL CERTAIN ARRAYS,SET COUNTERS ,AND DEFINE
C                        A CODE FOR EACH NODE OF THE SFG
       MPL=0
      KIK=1
      LIL=1
```

```
      DO 601 KAM=1,NEXPS
      DO 601 KIM=1,NTC
  601  POLY(KAM,KIM)=0
      DO 602  KP1=1,NEXPS
  602  MSCRT(KP1)=0
      DO 950 KO2=1,NSPT
  950  KODI(KO2)=0
      DO 603 KP2=1,NTC
  603  KSCRT(KP2)=0
      IR=1
      NFIR=1
        KNC=0
      KODES(1)=1
      DO 2000 JS=2,NNG
 2000  KODES(JS)=2*KODES(JS-1)
      IF(LISTP)175,175,1116
 1116  WRITE(6,170)NIN,NOUT
  170  FORMAT(* PATHS FROM NODE *,I2,* TO NODE *,I2//)
      WRITE(6,1905)
 1905 FORMAT(5X,*NO.      NODE LIST*)
  175  CONTINUE
      IF(LISTP)1113,1113,23
 1113  K3=LT(NIN)+1
       N(NIN,K3)=0
      K2=LT(1)+1
      N(1,K2)=-1
      NIN=1
      NOUT=1
      KLAS=0
   24  NFIR=0
      IF(LISTC)1209,1209,1219
 1219  CONTINUE
      WRITE(6,177)
  177  FORMAT(1X,*CIRCUITS*//)
      WRITE(6,1905)
        KNC=0
 1209  CONTINUE
C                     PROGRAM MAIN--4
C                      PATH -FINDING ALGORITHM
C                      IN ADDITION,STEP PF7 CALCULATES THE COMPOSITE
C                      CODE ,CONSTANT,AND EXPONENT OF THE PATH
C     PF1(PRELIMINARY)
      DO   1112 IZO=1,NNG
 1112  IFLOW(IZO)=0
      DO 31 I1=1,NNG
   31  KONC(I1)=1
        NOP=KLAS
      KLAS=0
   23  I=2
      J=NIN
```

```
      NP(1)=NIN
      IFLOW(NIN)=1
      IFLOW(NOUT)=-1
C
  25  K=KONC(J)
C
C  PF2(FIND NEXT NODE)
      NP(I)=N(J,K)
C
C     PF3 (TEST ROUTING MATRIX)
      IF(N(J,K))100,60,34
C
C     PF4(TEST FOR FLOWER)
  34  NJK=N(J,K)
      IF(IFLOW(NJK))50,38,26
  26  KONC(J)=KONC(J)+1
      GO TO 25
C     PF5(PREPARE FOR NEXT NODE)
  38   J=NP(I)
      IFLOW(J)=1
      I=I+1
      GO TO 25
C
C     PF6(BACKSTEP)
  60  IFLOW(J)=0
      KONC(J)=1
      J=NP(I-2)
      KONC(J)=KONC(J)+1
      I=I-1
      GO TO 25
C
C    PF7(FINISH PATH)
  50  KONC(J)=KONC(J)+1
      KLAS=KLAS+1
C
C    FIND CODE FOR NODE PATH
      NPCODE(IR)=0
      ISU=I-1
      DO 2002 IS=1,ISU
      NODS=NP(IS)
 2002  NPCODE(IR)=NPCODE(IR)+KODES(NODS)
C
C     CALL ARRAY AND WRITE
      IF(NFIR.EQ.1)GO TO 179
      IF(LISTC)1208,1208,1206
 1206  CONTINUE
      KRU=I
  179 KNO=KNO+1
      WRITE(6,110)KNO,(NP(KR),KR=1,KRU)
  110  FORMAT(4X,I3,6X,35I3)
```

```
 1208  CONTINUE
C
      IF(NFIR.EQ.1)GO TO 320
      KODET(IR)=0
      CONST(IR)=1.
      IXPOT(IR)=0
      IEND=I
      DO 319 KEW=2,IEND
      JNODE=NP(KEW-1)
      LNODE=NP(KEW)
      KODET(IR)=KODET(IR)+KODE(JNODE,LNODE)
      CONST(IR)=CONST(IR)*CONS(JNODE,LNODE)
      IXPOT(IR)=IXPOT(IR)+IXPO(JNODE,LNODE)
  319  CONTINUE
      CONEW=CONST(IR)
      IXNEW=IXPOT(IR)
      KONEW=KODET(IR)
      CALL ARRAY(1,CONEW,IXNEW,KONEW,POLY,LIL,KIK)
  320  CONTINUE
C
C
      IR=IR+1
      IF(IR-NPAC)1361,1361,1360
 1360  WRITE(6,1362)
 1362 FORMAT(1X,* NO. OF CIRCUITS EXCEEDS LIMIT-INCREASE DIMENSION*/
     1*CONTANING NPAC*)
 1361   CONTINUE
      GO TO 25
C
C
C                         PROGRAM MAIN--5
C     MODIFY THE SFG BY REMOVING EVERY BRANCH CONNECTED TO THE NODE THRO
C     WHICH ALL CIRCUITS HAVE JUST BEEN FOUND
  100 T3=0.
      IF(NCIR-1)2010,102,2010
  102 CONTINUE
      IF(NFIR-1)104,2010,104
  103 K4=LT(NIN)+1
      N(NIN,K4)=0
      K5=LT(1)+1
      N(1,K5)=-1
      NIN=1
      NOUT=1
      GO TO 24
  104 IF(NIN-JLAS) 105,200,200
  105 NIN=J+1
      NOUT=J+1
      KONC(J)=1
      NY=LT(J)+1
      N(J,NY)=0
```

```
      DO 109 JC=NIN,JLAS
      LCOL=LT(JC)
      IF(LCOL.EQ.0)GO TO 109
      IF(N(JC,LCOL)-J)109,107,109
  107 N(JC,LCOL)=0
      LT(JC)=LT(JC)-1
  109 CONTINUE
      NZ=LT(NIN)+1
      N(NIN,NZ)=-1
      NCUT=NIN
      GO TO 23
 2010 IF(NCIR-1)250,103,250
  200 CONTINUE
      NCL=KLAS
C     PROGRAM MAIN--6
C     FIND SECOND ORDER LOOPS
      NCL=KLAS
      KHOL=0
      DO 257 KOW=1,NPAC
  257 NOCTOT(KOW)=0
      LOW1=NCP+1
      NOC=0
      NCL1=NCL-1
      DO 203 LIR1=LOW1,NCL1
      LOW2=LIR1+1
      DO 202 LIR2=LOW2,NCL
      CALL IAND(NPCODE(LIR1),NPCODE(LIR2),NAN,0,KBASIS)
      IF(NAN)202,201,202
  201 CONTINUE
      TCONS2=CONST(LIR1)*CONST(LIR2)
      KXPO2=IXPOT(LIR1)+IXPOT(LIR2)
      KSYM2=KODET(LIR1)+KODET(LIR2)
      CALL ARRAY(2,TCONS2,KXPO2,KSYM2,POLY,LIL,KIK)
      KHOL=KHOL+1
      NOC=NOC+1
      IF(NOC-NEON)1396,1396,1395
 1395 WRITE(6,1397)
 1397 FORMAT(1X,*INCREASE NEON-THE DIMENSION OF THE ARRAY NOTCH*)
 1396 CONTINUE
      NOTCH(NOC)=LIR2
  202 CONTINUE
  203 NOCTOT(LIR1)=NOC
      NOCTOT(NCL)=NOC
C     PROGRAM MAIN  7
C     FIND LOOPS OF ORDER GREATER THAN 2
C    GENERATE THE FIRST ROW OF ISET
      NIPL=NCP+1
      KAPMAX=1
      INKO=1
      DO 1170 ISC=NIPL,NOL
```

```
      INK1=NCCTOT(ISC)                                                  SNA04420
      IF(ISC-1)1171,1171,1172                                           SNA04430
 1172 INK2=NCCTOT(ISC-1)+1                                              SNA04440
      GO TO 1173                                                        SNA04450
 1171 INK2=1                                                            SNA04460
 1173 IF(INK1-INK2-INKO)1170,1170,1175                                  SNA04470
 1175 INKO=INK1-INK2                                                    SNA04480
 1170 CONTINUE                                                          SNA04490
      IF(INKO-NCI)1391,1391,1390                                        SNA04500
 1390 WRITE(6,1392)INKO                                                 SNA04510
 1392 FORMAT(1X,*INCREASE NCI  THE NO OF COLUMNS IN DIMENSION OF ISET*) SNA04520
 1391 CONTINUE                                                          SNA04530
      DO 490 NIP=NIPL,NCL                                               SNA04540
      INKU=NCCTOT(NIP)                                                  SNA04550
      IF(NIP-1)210,210,211                                              SNA04560
  211 INKL=NCCTOT(NIP-1)+1                                              SNA04570
      GO TO 212                                                         SNA04580
  210 INKL=1                                                            SNA04590
  212 CONTINUE                                                          SNA04600
      IF(INKU-INKL)490,490,410                                          SNA04610
  410 JIP=0                                                             SNA04620
      DO 480 INK=INKL,INKU                                              SNA04630
      JIP=JIP+1                                                         SNA04640
  480 ISET(1,JIP)=NOTCH(INK)                                            SNA04650
      MAPC(NIP)=INKU-INKL+1                                             SNA04660
C     INITIATE PROCESS FOR FINDING HIGHER ORDER LOOPS                   SNA04670
      DO 430 KAT=1,NFAC                                                 SNA04680
      JAC(KAT)=0                                                        SNA04690
  430 NUP(KAT)=0                                                        SNA04700
      JAC(1)=MAPC(NIP)                                                  SNA04710
      KAP=2                                                             SNA04720
  440 KAP=KAP-1                                                         SNA04730
      IF(KAP)490,490,429                                                SNA04740
  425 KAP=KAP+1                                                         SNA04750
      IF(KAP-NRI)1350,1350,1351                                         SNA04760
 1351 WRITE(6,1352)                                                     SNA04770
 1352 FORMAT(1X,*INCREASE NRI- THE NO. OF ROWS IN DIMENSION OF ISET*)   SNA04780
 1350 CONTINUE                                                          SNA04790
      NUP(KAP)=0                                                        SNA04800
  429 KAP1=KAP+1                                                        SNA04810
      JAC(KAP1)=0                                                       SNA04820
      NUP(KAP)=NUP(KAP)+1                                               SNA04830
C     LABEL LOOP OF FIRST CIRCUIT                                       SNA04840
      NAP=NUP(KAP)                                                      SNA04850
      IF(KAPMAX-KAP)1347,1348,1348                                      SNA04860
 1347 KAPMAX=KAP                                                        SNA04870
 1348 CONTINUE                                                          SNA04880
      ISAT=ISET(KAP,NAP)                                                SNA04890
C     TEST LOOP OF REMAINING CKTS                                       SNA04900
      MAPU=JAC(KAP)                                                     SNA04910
```

```
      MAPL=NUP(KAP)+1
      DO 435 MAPI=MAPL,MAPU
      ISOT=ISET(KAP,MAPI)
      CALL IANC(NPCODE(ISAT),NPCODE(ISOT),KAN,0,KBASIS)
      IF(KAN)435,455,435
  455 CONTINUE
C     WRITE
      TCONSG=CONST(NIP)
      KXPOG=IXPOT(NIP)
      KSYMG=KODET(NIP)
      DO 477 LPO=1,KAP
      ITIC=NUP(LPO)
      ITUCH=ISET(LPO,ITIC)
      TCONSG=TCONSG*CONST(ITUCH)
      KXPOG=KXPOG+IXPOT(ITUCH)
  477 KSYMG=KSYMG+KODET(ITUCH)
      TCONSG=TCONSG*CONST(ISOT)
      KXPOG=KXPOG+IXPOT(ISOT)
      KSYMG=KSYMG+KODET(ISOT)
      KAPP=KAP+2
      CALL ARRAY(KAPP,TCONSG,KXPOG,KSYMG,POLY,LIL,KIK)
      KHOL=KHOL+1
C     SET COUNTERS
  423 KAP1=KAP+1
      JAC(KAP1)=JAC(KAP1)+1
      JACK=JAC(KAP1)
      ISET(KAP1,JACK)=ISET(KAP,MAPI)
  435 CONTINUE
      JACK=JAC(KAP1)
      IF(JACK-2) 431,425,425
  431 IF(JAC(KAP)-NUP(KAP)-1) 440,440,429
  490 CONTINUE
      CALL ARRAY(2,1.,0,0,POLY,LIL,KIK)
C     PROGRAM MAIN  8
C     DECODE COMPOSITE SYMBOL CODE
C     AND ISOLATE SYMBOLS FROM
C     INVERSE SYMBOLS
      NANU=LIL-1
      DO 691 J1=1,NEXPS
      DO 691 J2=1,NTC
  691 POLYU(J1,J2)=0
      DO 693 J1=1,NTC
      DO 693 J2=1,NSFTU
      SEMPON(J1,J2)=STAR(1)
      SEMPOD(J1,J2)=STAR(1)
      SIMBON(J1,J2)=SB
  693 SIMBOD(J1,J2)=SB
      DO 951 J4=1,NTC
      NA(J4)=0
  951 NB(J4)=0
```

```
C     DECODE KSORT(JZ) AND RECORD TERMS
C     CONTAINING FEEDBACK SYMBOL 'FB'
      JZU=LIL-1
      DO 646 JZ=1,JZU
      KODY=KSORT(JZ)
      ITOP(JZ)=0
      IF(KODY)715,646,715
  715 CALL DECODE(KOO,KODY,IZ,FB,JZ,SEMBOL,KODF,KODI,ITOP,KBASIS)
C      ISOLATE NUM. SYMBOLS AND INVERSE SYMBOLS
C      OF KSORT(JZ)
  637 NAK=0
      NAT=0
      IF(IZ)646,646,647
  647 CONTINUE
      DO 645 NZ=1,IZ
      KOZY=KODI(NZ)
      IARG=KODF(NZ)
      IF(IARG-NRS) 1340,1340,1341
 1341 WRITE(6,1342)
 1342 FORMAT(1X,*INCREASE THE DIMENSION OF STAR*)
 1340 CONTINUE
      IF(KONS(KOZY))657,657,659
  657 NAK=NAK+1
      IF(NAK-NSPTU-1)1376,1375,1375
 1375 WRITE(6,1377)
 1377 FORMAT(1X,*NSPT EXCEEDS LIMIT- INCREASE DIMENSIONS, CONTAINING
     1 NSPT*)
 1376 CONTINUE
      SIMBON(JZ,NAK)=SEMBOL(KOZY)
      SEMPON(JZ,NAK)=STAR(IARG)
      NA(JZ)=NA(JZ)+1
      GO TO 645
  659 NAT=NAT+1
      IF(NAT-NSPTU-1)1381,1380,1380
 1380 WRITE(6,1382)
 1382 FORMAT(1X,*NSPT EXCEEDS LIMIT-INCREASE DIMENSIONS,*
     1*CONTAINING NSPT*)
 1381 CONTINUE
      SIMBOD(JZ,NAT)=SEMBOL(KOZY)
      SEMPOD(JZ,NAT)=STAR(IARG)
      NB(JZ)=NB(JZ)+1
  645 CONTINUE
  646 CONTINUE
C     PROGRAM MAIN  9
C     SEPARATE POLY INTO ARRAYS FOR THE NUMERATOR AND DENOMINATOR
C     OF THE TRANSFER FUNCTION
C     THE CONSTANT COEFFICIENTS IN THE TRANSFER FUNCTION ARE SEPARATED
C     INTO ARRAYS FOR THE NUMERATOR AND DENOMINATOR
      KIKU=KIK-1
      DO 755 JA=1,KIKU
```

```
      JIB=0
      JD=0
      DO 755 JC=1,NANU
      IF(ITOP(JC))753,753,751
  751 JIB=JIB+1
      POLYU(JA,JIB)=POLY(JA,JC)
      GO TO 755
  753 JD=JD+1
      POLY(JA,JD)=POLY(JA,JC)
  755 CONTINUE
C     PROGRAM MAIN  10
C     MAKE POWERS OF S IN OUTPUT
C     TRANSFER FUNCTION POSITIVE
      MAXIM=0
      KARU=KIK-1
      DO 522 KAR=1,KARU
      IF(MSORT(KAR))521,522,522
  521 IF(MAXIM+MSORT(KAR))523,522,522
  523 MAXIM=-MSORT(KAR)
  522 CONTINUE
      DO 524 KIT=1,KARU
  524 MSORT(KIT)=MAXIM+MSORT(KIT)
C     MAIN PROGRAM  11
C     PRINT OUT NUMERATOR OF THE TRANSFER FUNCTION
      LUK=0
      IKU=LIL-1
      WRITE(6,931)
      WRITE(6,930)
      WRITE(6,920)
  920 FORMAT(25X,*NUMERATOR POLYNOMIAL*///)
      WRITE(6,921)
  921 FORMAT(1X,*COLUMN*,12X,*SYMBOL FOR GIVEN COLUMN*)
      DO 905 IK=1,IKU
      IF(ITOP(IK))905,905,901
  901 ILU=NA(IK)
      IF(ILU)710,710,711
  710 ILU=1
  711 JLU=NB(IK)
      IF(JLU)712,712,713
  712 JLU=1
  713 CONTINUE
      LUK=LUK+1
      WRITE(6,903)LUK,(SIMBON(IK,IL),SEMPON(IK,IL),
     1IL=1,ILU),DASH,(SIMBOD(IK,JL),SEMPOD(IK,JL),JL=1,JLU)
  903 FORMAT(1X,I5,20X,30A3)
  905 CONTINUE
      WRITE(6,930)
  930 FORMAT(//)
      WRITE(6,1821)
 1821 FORMAT(1X,*POWER*)
```

```
      WRITE(6,922)
  922 FORMAT(1X,*OF S*,17X,*CONSTANT COEFS, IN THE POLYNOMIAL*)
      LML=1
        LMU=4
      IF(JIB-LMU)820,818,818
  820 LMU=JIB
  818 WRITE(6,806)(LC,LC=LML,LMU)
  806 FORMAT(2X,7(8X,*COLUMN*,I2))
      KROWU=KIK-1
      DO 808 KROW=1,KROWU
      WRITE(6,810)MSORT(KROW),(POLYU(KROW,LM),LM=LML,LMU)
  810 FORMAT(I5,*    *,7(E12.5,*    *))
  808 CONTINUE
      IF(JIB-LMU)814,814,812
 812   LML=LML+4
       LMU=LMU+4
      IF(JIB-LMU)816,818,818
  816 LMU=JIB
      GO TO 818
  814 CONTINUE
C     PROGRAM MAIN  12
C     PRINT OUT DENOMINATOR OF
C     THE TRANSFER FUNCTION
      LUK=0
      IKU=LIL-1
      WRITE(6,931)
  931 FORMAT(1X,50(1H*))
      WRITE(6,930)
      WRITE(6,923)
  923 FORMAT(25X,*DENOMINATOR POLYNOMIAL*///)
      WRITE(6,924)
  924 FORMAT(1X,*COLUMN*,12X,*SYMBOL FOR GIVEN COLUMN*)
      DO 705 IK=1,IKU
      IF(ITOP(IK))701,701,705
  701 ILU=NA(IK)
      LUK=LUK+1
      IF(ILU)915,915,916
  915 ILU=1
  916 JLU=NB(IK)
      IF(JLU)917,917,918
  917 JLU=1
  918 CONTINUE
      WRITE(6,703) LUK,(SIMBON(IK,IL),SEMPON(IK,IL),
     1IL=1,ILU),DASH,(SIMBOD(IK,JL),SEMPOD(IK,JL),JL=1,JLU)
  703 FORMAT(1X,I5,20X,30A3)
  705 CONTINUE
      WRITE(6,930)
      WRITE(6,1822)
 1822 FORMAT(1X,*POWER*)
      WRITE(6,925)
```

```
 525 FCRMAT(1X,*  CF S  *,17X,*CONSTANT COEFS.IN THE POLYNOMIAL*)          SNA06920
     LML=1                                                                SNA06930
      LML=4                                                               SNA06940
     IF(JC-LML)520,518,518                                                SNA06950
 520 LMU=JC                                                               SNA06960
 518 WRITE(6,506) (LC,LC=LML,LML)                                         SNA06970
 506 FCRMAT(2X,7(8X,*CCLUMN*,I2))                                         SNA06980
     KROWL=KIK-1                                                          SNA06990
     DC 508 KROW=1,KROWU                                                  SNA07000
     WRITE(6,510) MSCRT(KRCW),(PCLY(KROW,LM),LM=LML,LMU)                  SNA07010
 510 FCRMAT(I5,*   *,7(E12.5,*    *))                                     SNA07020
 508 CCNTINUE                                                             SNA07030
     IF(JC-LML)514,514,512                                                SNA07040
 512  LML=LML+4                                                           SNA07050
      LML=LMU+4                                                           SNA07060
     IF(JC-LML)516,518,518                                                SNA07070
 516 LMU=JC                                                               SNA07080
     GC TO 518                                                            SNA07090
 514 CCNTINUE                                                             SNA07100
     WRITE(6,930)                                                         SNA07110
 250 GC TO 1111                                                           SNA07120
     END                                                                  SNA07130
C                                                                         SNA07140
     SUBROUTINE SFG(NFIRST,NLAST,IXPCN,WEIGT,SYMBUL,KONSO,MIX,NEST,       SNA07150
    1LIST,NIN,NOUT,NOD,NCB,LISTG,NCDA,NODB)                               SNA07160
C*************************************************************************SNA07170
C  THE FOLLCWING ARRAYS ARE ASSOCIATED WITH THE NETWORK                   SNA07180
C  CHARACTERISTIC NBN (DEFINED IN PROGRAM MAIN-1)                         SNA07190
     DIMENSION JRCW(25),NP(25),IVV(25),NUML(25),ICV(25),INTREE(25)        SNA07200
     DIMENSION LINC(25)                                                   SNA07210
     DIMENSICN NF(25,25),IE(25,25),NS(25,25)                              SNA07220
     DIMENSION TYPB(25),JB(25),LB(25),MSYM(25)                            SNA07230
     DIMENSION IQUAL(25),VAL(25),SYM(25)                                  SNA07240
     DIMENSION IQUALX(25),VALX(25),NUMLX(25),INTRE(25),NOTREE(25)         SNA07250
     DIMENSICN TYPX(25),NUMX(25),JBX(25),LBX(25),SYMX(25)                 SNA07260
C*************************************************************************SNA07270
C  THE FOLLOWING ARRAYS ARE ASSOCIATED WITH THE NETWORK                   SNA07280
C  CHARACTERISTIC NBG                                                     SNA07290
     DIMENSION NFIRST(75),NLAST(75),IXPON(75),WEIGT(75)                   SNA07300
     DIMENSION KCNSC(75),NEST(75),TYPE(75),MAPY(75)                       SNA07310
     DIMENSION SYMBUL(75),MIX(75),CVAL(75)                                SNA07320
      DIMENSION SM(6),SN(6)                                               SNA07321
C*************************************************************************SNA07330
C                                                                         SNA07340
     CCMMON/C2/NNG,NBG                                                    SNA07350
      COMMON/C5/SM,SN,NCCUT,NINN                                          SNA07351
C                                                                         SNA07360
C       SUBPROGRAM 'A'                                                    SNA07370
     DATA Y,G,C,IC,R,CL,Z/2HY ,2HG ,2HC ,1H=,2HR ,2HL ,2HZ /              SNA07380
     DATA E,CI,CC,CV,VV,VC/2HE ,2HI ,2HCC,2HCV,2HVV,2HVC/                 SNA07390
```

```
      CATA FB/3H FB/                                                    SNA07400
      CATA CNE/3H  1/                                                   SNA07410
      CC 71C IC=1,NNG                                                   SNA07420
      CC 71C IK=1,NNG                                                   SNA07430
      NS(IC,IK)=0                                                       SNA07440
  710 NF(IC,IK)=0                                                       SNA07450
      LINK=C                                                            SNA07460
      CC 152 IG=1,NBG                                                   SNA07470
      NEST(IG)=0                                                        SNA07480
  152 KCNSC(IG)=0                                                       SNA07490
      WRITE (6,26C)                                                     SNA07500
  260 FCRMAT (//)                                                       SNA07510
C                                                                       SNA07520
C                                                                       SNA07530
C                                                                       SNA07540
      IXPCN(1)=0                                                        SNA07550
      WEIGT(1)=-1.                                                      SNA07560
      SYMBUL(1)=FB                                                      SNA07570
      KCNSC(1)=0                                                        SNA07580
      NEST(1)=1                                                         SNA07590
      MC=C                                                              SNA07600
      LC=C                                                              SNA07610
      LIST=1                                                            SNA07620
      KLU=0                                                             SNA07630
      CC 5 I1=1,NNG                                                     SNA07640
      INTREE(I1)=C                                                      SNA07650
    5 JROW(I1)=0                                                        SNA07660
      CC 528 I=1,NCB                                                    SNA07670
      READ(5,9) TYPX(I),NUMX(I),JBX(I),LBX(I),SYMX(I),                  SNA07680
     1ICUALX(I),VALX(I),NUMLX(I)                                        SNA07690
      IF(TYPX(I).EC.CC)GC TC 1300                                       SNA07700
      IF(TYPX(I).EC.CV)GC TC 1300                                       SNA07710
      IF(TYPX(I).EC.VV)GC TC 1300                                       SNA07720
      IF(TYPX(I).EC.VC)GC TC 1300                                       SNA07730
      GC TO 1301                                                        SNA07740
 1300 IF(NUMLX(I))1301,1302,1301                                        SNA07750
 1302 WRITE(6,1303)                                                     SNA07760
 1303 FCRMAT(1X,49H***ERRCR***CCNTRCL SPECIFICATION FOR CEP. SOURCE ,   SNA07770
     17HMISSING)                                                        SNA07771
      GC TO 7000                                                        SNA07772
 1301 CCNTINUE                                                          SNA07773
  528 CCNTINUE                                                          SNA07774
      GC TC 7777                                                        SNA07775
 7000 NCB=0                                                             SNA07776
       GO TC 1305                                                       SNA07777
    9 FCRMAT (A2,I3,2I5,1X,A3,A1,E12.5,I3)                              SNA07778
 7777 CONTINUE                                                          SNA07779
       KJ=0                                                             SNA07780
      MMM=1                                                             SNA07781
      IF(NINN-1)222,222,3333                                            SNA07782
```

```
3333   CONTINUE                                                         SNA07783
       KJ=1                                                             SNA07784
 222  READ(5,224)NIN,K                                                  SNA07785
 224   FORMAT(2I5)                                                      SNA07786
      WRITE(6,225)NIN                                                   SNA07788
 225   FORMAT(1X,*ELEMENT NO. OF SOURCE =*I3)                           SNA07789
      IF(KJ)936,936,937                                                 SNA07790
 937   N=NINN-1                                                         SNA07791
      DO 226 I=1,N                                                      SNA07793
      READ(5,227)NI,M                                                   SNA07794
 227   FORMAT(2I5)                                                      SNA07795
       WRITE(6,928)I,NI                                                 SNA07796
 928  FORMAT(1X,*ELEMENT NO. OF SOURCE (*,I2,*)=*,I3)                   SNA07798
      IF(K)929,929,930                                                  SNA07799
 929  IF(M)931,931,932                                                  SNA07800
 931  TYPX(NI)=VV                                                       SNA07801
      GO TO 935                                                         SNA07802
 932   TYPX(NI)=VC                                                      SNA07803
      GO TO 935                                                         SNA07804
 930  IF(M)933,933,934                                                  SNA07805
 933   TYPX(NI)=CV                                                      SNA07806
      GO TO 935                                                         SNA07807
 934  TYPX(NI)=CC                                                       SNA07809
 935   SYMX(NI)=SN(MMM)                                                 SNA07810
      NUMLX(NI)=NIN                                                     SNA07812
      MMM=MMM+1                                                         SNA07815
226    CONTINUE                                                         SNA07816
 936  CONTINUE                                                          SNA07817
       KKK=1                                                            SNA07818
 5559 CONTINUE                                                          SNA07819
      WRITE(6,260)                                                      SNA07820
       IF(KKK-1)2603,2602,2603                                          SNA07821
2602   WRITE(6,2600)                                                    SNA07822
 2600  FORMAT(30X,*NETWORK*)                                            SNA07823
       GO TO 2604                                                       SNA07824
 2603  WRITE(6,2601)                                                    SNA07826
2601   FORMAT(30X,*MODIFIED NETWORK*)                                   SNA07827
 2604 CONTINUE                                                          SNA07828
      WRITE(6,261)                                                      SNA07829
 261  FORMAT(1X,*ELEMENT ELEMENT INTIAL TERMINAL ELEMENT ELEMENT ELEMENTSNA07830
     1 NO.*)                                                            SNA07832
C                                                                       SNA07840
C                                                                       SNA07850
C                                                                       SNA07860
C                                                                       SNA07870
C                                                                       SNA07880
      WRITE(6,262)                                                      SNA07890
 262  FORMAT(1X,* TYPE   NUMBER   NODE     NODE   SYMBOL   VALUE  OF CONSNA07900
     1TROL*)                                                            SNA07910
      DO 601 M=1,NCB                                                    SNA07920
```

```
  601 WRITE(6,600) TYPX(M),NUMX(M),JBX(M),LBX(M),SYMX(M),          SNA07930
     1ICUALX(M),VALX(M),NUMLX(M)                                    SNA07940
  600 FORMAT(4X,A2,6X,I2,6X,I2,6X,I2,6X,A3,A1,E12.5, 2X,I2)        SNA07950
      CALL FTREE(TYPX,JBX,LBX,INTRE,NCTREE,NCD,NOB)                 SNA07960
       KLL=0                                                        SNA07961
C                                                                   SNA07970
C      SUBPROGRAM 'B'                                               SNA07980
      WRITE(6,518)                                                  SNA07990
  518 FORMAT(30X,13HTREE SELECTED)                                  SNA08000
      NUML=NCD-1                                                    SNA08010
      DO 21 NU=1,NUML                                               SNA08020
      IC=INTRE(NU)                                                  SNA08030
      NUMC=NUMX(IC)                                                 SNA08040
      TYPB(NUMC)=TYPX(IC)                                           SNA08050
      JB(NUMC)=JBX(IC)                                              SNA08060
      LB(NUMC)=LBX(IC)                                              SNA08070
      SYM(NUMC)=SYMX(IC)                                            SNA08080
      ICUAL(NUMC)=ICUALX(IC)                                        SNA08090
      VAL(NUMC)=VALX(IC)                                            SNA08100
      NUML(NUMC)=NUMLX(IC)                                          SNA08110
      INTREE(NUMC)=1                                                SNA08120
      WRITE(6,517)TYPB(NUMC),NUMC,JB(NUMC),LB(NUMC),SYM(NUMC),      SNA08130
     1ICUAL(NUMC),VAL(NUMC),NUML(NUMC)                              SNA08140
  517 FORMAT (4X,A2,6X,I2,6X,I2,6X,I2,6X,A3,A1,E12.5,2X,I2)         SNA08150
      KLU=KLL+1                                                     SNA08160
      LINC(NUMC)=0                                                  SNA08170
      IF(TYPB(NUMC).NE.VV) GO TO 3                                  SNA08180
      MO=MO+1                                                       SNA08190
      IVV(MO)=NUMC                                                  SNA08200
    3 IF(TYPB(NUMC).NE.CV) GO TO 4                                  SNA08210
      LO=LO+1                                                       SNA08220
      ICV(LO)=NUMC                                                  SNA08230
    4 JF=JB(NUMC)                                                   SNA08240
      LF=LB(NUMC)                                                   SNA08250
      IB(JF,LF)=NUMC                                                SNA08260
      IB(LF,JF)=NUMC                                                SNA08270
      JROW(JF)=JROW(JF)+1                                           SNA08280
      JROJ=JROW(JF)                                                 SNA08290
      NF(JF,JROJ)=LF                                                SNA08300
      NS(JF,LF)=1                                                   SNA08310
      JROW(LF)=JROW(LF)+1                                           SNA08320
      JROL=JROW(LF)                                                 SNA08330
      NF(LF,JROL)=JF                                                SNA08340
      NS(LF,JF)=-1                                                  SNA08350
   21 CONTINUE                                                      SNA08351
       IF(KKK-1)6660,6661,6660                                      SNA08352
 6661  CONTINUE                                                     SNA08353
       IF(NOCUT-1)8000,211,8000                                     SNA08354
 8000  NODA=NOD                                                     SNA08355
       MM=1                                                         SNA08356
```

```
22     READ(5,12)NOOUT,NODAA,NODBB,K                                          SNA08357
        WRITE(6,260)                                                          SNA08358
        IF(NOOUT)5561,5560,5561                                               SNA08359
5561    WRITE(6,5562)KJ,NOOUT                                                 SNA08360
 5562   FORMAT(1X,*ELEMENT NUMBER ASSOCIATED WITH OUTPUT(*,I1,*)=*,I3)        SNA08361
        GO TO 5565                                                            SNA08362
5560    WRITE(6,5563)KJ,NODAA                                                 SNA08363
 5563   FORMAT(1X,*POSITIVE OUTPUT VOLTAGE TERMINAL(*,I1,*)=*I3)              SNA08364
        WRITE(6,5564)KJ,NODBB                                                 SNA08365
5564    FORMAT(1X,*NEGATIVE OUTPUT VOLTAGE TERMINAL(*,I1,*)=*,I3)             SNA08366
5565    KJ=KJ+1                                                               SNA08367
       IF(NOOUT)113,113,14                                                    SNA08368
14     NOB=NOB+1                                                              SNA08369
       IF(K) 15,15,16                                                         SNA08370
 15    TYPX(NOB)=VV                                                           SNA08371
       GO TO1117                                                              SNA08372
 16    TYPX(NOB)=CV                                                           SNA08374
1117    NUMX(NOB)=NOB                                                         SNA08375
        JBX(NOB)=NOD                                                          SNA08376
       LBX(NOB)=NOD+1                                                         SNA08377
       SYMX(NOB)=SM(MM)                                                       SNA08379
       MM=MM+1                                                                SNA08380
         NUMLX(NOB)=NOOUT                                                     SNA08381
        NOD=NOD+1                                                             SNA08383
        GO TO 17                                                              SNA08385
113    CALL TREP (NODAA,NODBB,NF,NP,NPL)                                      SNA08386
       NPLL=NPL-1                                                             SNA08387
       DO 18 I=1,NPLL                                                         SNA08388
        NOB=NOB+1                                                             SNA08389
        TYPX(NOB)=VV                                                          SNA08390
       NUMX(NOB)=NOB                                                          SNA08391
        JBX(NOB)=NOD                                                          SNA08393
        LBX(NOB)=NOD+1                                                        SNA08394
       SYMX(NOB)=SM(MM)                                                       SNA08395
        NP1=NP(I)                                                             SNA08396
        NP2=NP(I+1)                                                           SNA08397
       NUMLX(NOB)=IB(NP1,NP2)                                                 SNA08398
        NOD=NOD+1                                                             SNA08399
 18    CONTINUE                                                               SNA08400
         MM=MM+1                                                              SNA08401
 17    NODB=NOD                                                               SNA08402
        IF(NOOUT-1)20,20,8001                                                 SNA08403
 8001  NOOUT=NOOUT-1                                                          SNA08404
        GO TO 22                                                              SNA08405
 211   READ(5,12) NOUT,NODA,NODB,K                                            SNA08406
        IF(NOUT)5550,5551,5550                                                SNA08407
 5550   WRITE(1,5555)NOUT                                                     SNA08408
 5555   FORMAT(1X,*ELEMENT NUMBER ASSOCIATED WITH OUTPUT=*,I3)                SNA08409
        GO TO 6660                                                            SNA08410
5551   WRITE(6,5556)NODA                                                      SNA08411
```

```
5556   FORMAT(1X,*POSITIVE OUTPUT VOLTAGE TERMINAL=*,I3)               SNA08412
       WRITE(6,5557)NCDB                                               SNA08413
5557   FORMAT(1X,*NEGATIVE OUTPUT VOLTAGE TERMINAL=*,I3)               SNA08414
       GO TO 6660                                                      SNA08415
 20    KKK=KKK-1                                                       SNA08416
       GO TO 5559                                                      SNA08417
 12    FORMAT(4I5)                                                     SNA08418
 6660 DO 13 ILL=1,NOD                                                  SNA08420
      JROI=JROW(ILL)+1                                                 SNA08421
   13 NF(ILL,JROI)=0                                                   SNA08422
      WRITE(6,260)                                                     SNA08423
      WRITE(6,715)                                                     SNA08427
 715   FORMAT(30X,* SFG *,/)                                           SNA08428
C                                                                      SNA08430
C      SUBPROGRAM 'C'                                                  SNA08440
C      THIS PROGRAM GENERATES SIGNAL FLOW GRAPH INFO.                  SNA08450
C      FROM BRANCH NODE TO LINK NODE                                   SNA08460
      NCBY=NCB                                                         SNA08470
  151 CONTINUE                                                         SNA08480
      NES=0                                                            SNA08490
      LON=0                                                            SNA08500
      IF(KLU-NCB)532,360,532                                           SNA08510
  532 LINK=LINK+1                                                      SNA08520
      IF(NOTREE(LINK))534,534,532                                      SNA08530
  534 NUMC=NUMX(LINK)                                                  SNA08540
      TYPE(NUMC)=TYPX(LINK)                                            SNA08550
      JK=JBX(LINK)                                                     SNA08560
      LK=LBX(LINK)                                                     SNA08570
      SYM(NUMC)=SYMX(LINK)                                             SNA08580
      IQUAL(NUMC)=IQUALX(LINK)                                         SNA08590
      CVAL(NUMC)=VALX(LINK)                                            SNA08600
      NUMB=NUMLX(LINK)                                                 SNA08610
      TYP2=TYPE(NUMC)                                                  SNA08620
      CVALL=CVAL(NUMC)                                                 SNA08630
      KLU=KLU+1                                                        SNA08640
      LINC(NUMC)=1                                                     SNA08650
      KDEPS=0                                                          SNA08660
      KANSO=0                                                          SNA08670
      IF(TYPE(NUMC).EQ.CL)GO TO 117                                    SNA08680
      IF(TYPE(NUMC).EQ.G)GO TO 119                                     SNA08690
      IF(TYPE(NUMC).EQ.Y)GO TO 119                                     SNA08700
      IF(TYPE(NUMC).EQ.R)GO TO 700                                     SNA08710
      IF(TYPE(NUMC).EQ.Z)GO TO 700                                     SNA08720
      IF(TYPE(NUMC).EQ.C)GO TO 121                                     SNA08730
      KDEPS=1                                                          SNA08740
      IF(TYPE(NUMC).EQ.E)GO TO 123                                     SNA08750
      IF(TYPE(NUMC).EQ.CI)GO TO 123                                    SNA08760
      IF(TYPE(NUMC).EQ.VC)GO TO 165                                    SNA08770
      IF(TYPE(NUMC).EQ.CC)GO TO 265                                    SNA08780
  117 IXPS=-1                                                          SNA08790
```

```
      KANSO=1
      GO TO 123
  119 IXPS=0
      GO TO 123
  700 IXPS=0
      KANSO=1
      GO TO 123
  121 IXPS=1
  123 CALL TREP(JK,LK,NF,NP,NPL)
      IFIN=NUMC
  149 LCN=LCN+1
      NP1=NP(LCN)
      NP2=NP(LCN+1)
  107 INIT=IB(NP1,NP2)
  109 SIGH=NS(NP1,NP2)
      IF(KDEPS)167,167,169
  167 IF(IQUAL(NUMC).EQ.IO)GO TO 111
      NES=1
      CONST=SIGH
      GO TO 125
  111 CONST=SIGH*CVALU
  125 LIST=LIST+1
      IF(NES)502,503,502
  502 NEST(LIST)=1
  503 KONSO(LIST)=KANSO
      NFIRST(LIST)=INIT
      NLAST(LIST)=IFIN
      SYMBUL(LIST)=SYM(IFIN)
      IXPON(LIST)=IXPS
      IF(KONSO(LIST))505,505,504
  504 WEIGT(LIST)=1./CONST
      GO TO 506
  505 WEIGT(LIST)=CONST
  506 MAPY(NUMC)=LIST
  127 FORMAT(3I5,E12.5)
  129 FORMAT (A4)
C
C      SUBPROGRAM 'D'
C      THIS PROGRAM GENERATES SIGNAL FLOW GRAPH INFO.
C      FROM LINK NODE TO BRANCH NODE
  169 CONTINUE
      IF(TYPB(INIT).EQ.E)GO TO 201
      IF(TYPB(INIT).EQ.CI)GO TO 201
      IF(TYPB(INIT).EQ.VV)GO TO 201
      IF(TYPB(INIT).EQ.CV)GO TO 201
      LIST=LIST+1
      IF(TYPB(INIT).EQ.R)GO TO 133
      IF(TYPB(INIT).EQ.Z)GO TO 133
      IF(TYPB(INIT).EQ.G)GO TO 702
      IF(TYPB(INIT).EQ.Y)GO TO 702
```

```
      IF(TYPE(INIT).EQ.CL)GO TO 135
      IF(TYPE(INIT).EQ.C)GO TO 137
  133 IXPON(LIST)=0
      GO TO 141
  702 IXPON(LIST)=0
      KONSO(LIST)=1
      GO TO 141
  135 IXPON(LIST)=1
      GO TO 141
  137 IXPON(LIST)=-1
      KONSO(LIST)=1
  141 IF(IQUAL(INIT).EQ.IC)GO TO 139
      NEST(LIST)=1
      WEIGT(LIST)=-1.*SIGH
      GO TO 147
  139 IF(KONSO(LIST))608,608,607
  607 WEIGT(LIST)=-SIGH/VAL(INIT)
      GO TO 147
  608 WEIGT(LIST)=-SIGH*VAL(INIT)
  147 NFIRST(LIST)=IFIN
      NLAST(LIST)=INIT
      SYMBUL(LIST)=SYM(INIT)
  201 NPLA=NPL-1-LON
      IF(NPLA)151,151,149
C
C       SUBPROGRAM 'E'
C       THIS PROGRAM SETS UP SFG INFO. FOR VC
C       TYPE CONTROL SOURCES
  165 NUNC=NUMB
      IF(INTREE(NUMB))163,163,161
  163 LIST=LIST+1
      NFIRST(LIST)=NUMB
      NCBY=NCBY+1
      NLAST(LIST)=NCBY
      SYMBUL(LIST)=SYM(NUMB)
      NUNO=NCBY
      IF(TYPE(NUMB).EQ.Y)GO TO 912
      IF(TYPE(NUMB).EQ.G)GO TO 912
      IF(TYPE(NUMB).EQ.C)GO TO 914
      IF(TYPE(NUMB).EQ.CL)GO TO 916
      KUNC=0
      KONSO(LIST)=0
      IXPON(LIST)=0
      GO TO 918
  912 IXPON(LIST)=0
      KONSO(LIST)=1
      KUNC=1
      GO TO 918
  914 IXPON(LIST)=-1
      KONSO(LIST)=1
```

```
      KUNC=1
      GO TO 918
  916 IXPON(LIST)=1
      KONSO(LIST)=0
      KUNC=0
  918 IF(IQUAL(NUMB).EQ.IO)GO TO 920
      NEST(LIST)=1
      WEIGT(LIST)=1.
      GO TO 209
  920 IF(KUNC)922,922,924
  922 WEIGT(LIST)=CVAL(NUMB)
      GO TO 209
  924 WEIGT(LIST)=1./CVAL(NUMB)
  209 CONTINUE
  161 LIST=LIST+1
      NFIRST(LIST)=NUNC
      NLAST(LIST)=NUMC
      SYMBUL(LIST)=SYM(NUMC)
      IXPON(LIST)=0
      IF(IQUAL(NUMC).EQ.IO)GO TO 171
      NEST(LIST)=1
      WEIGT(LIST)=1.
      GO TO 203
  171 WEIGT(LIST)=CVALU
  203 CONTINUE
      GO TO 123
C
C       SUBPROGRAM 'F'
C       THIS PROGRAM SETS UP SFG INFO. FOR CC
C       TYPE CONTROL SOURCES
  265 MUNC=NUMB
      IF(INTREE(NUMB))621,621,620
  620 LIST=LIST+1
      NFIRST(LIST)=NUMB
      NCBY=NCBY+1
      NLAST(LIST)=NCBY
      SYMBUL(LIST)=SYM(NUMB)
      MUNC=NCBY
      IF(TYPE(NUMB).EQ.Z)GO TO 233
      IF(TYPE(NUMB).EQ.R)GO TO 233
      IF(TYPE(NUMB).EQ.CL)GO TO 235
      IF(TYPE(NUMB).EQ.C)GO TO 237
      KUNC=0
      KONSO(LIST)=0
      IXPON(LIST)=0
      GO TO 241
  233 IXPON(LIST)=0
      KONSO(LIST)=1
      KUNC=1
      GO TO 241
```

```
 235 IXPCN(LIST)=-1
     KCNSC(LIST)=1
     KLNC=1
     GC TC 241
 237 IXPCN(LIST)=1
     KCNSC(LIST)=C
     KLNC=C
 241 IF(IQLAL(NUMB).EC.IC)GC TC 239
     NEST(LIST)=1
     WEIGT(LIST)=1
     GC TC 247
 239 IF(KUNC)900,90C,902
 9C0 WEIGT(LIST)=VAL(NUMB)
     GC TC 247
 9C2 WEIGT(LIST)=1./VAL(NUMB)
 247 CCNTINLE
 621 LIST=LIST+1
     NFIRST(LIST)=MLNC
     NLAST(LIST)=NUMC
     SYMBUL(LIST)=SYM(NUMC)
     IXPCN(LIST)=0
     IF(IQLAL(NUMC).EC.IC)GC TO 271
     NEST(LIST)=1
     WEIGT(LIST)=1.
     GC TC 281
 271 WEIGT(LIST)=CVALU
 281 CCNTINLE
     GC TO 123
C
C     SUBPRCGRAM 'G'
C     THIS PRCGRAM SETS UP SFG INFC. FOR VV
C     TYPE CONTRCL SCURCES
 360 IF(MO)460,460,364
 364 DC 305 MI=1,MC
     KI=IVV(MI)
     NUNO=NUML(KI)
     IF(LINC(NUNO))361,361,363
 363 LIST=LIST+1
     NFIRST(LIST)=NUML(KI)
     NCBY=NOBY+1
     NLAST(LIST)=NCBY
     SYMBUL(LIST)=SYM(NUNC)
     IF(TYPE(NUNO).EQ.Y)GC TO 333
     IF(TYPE(NUNO).EQ.G)GC TC 333
     IF(TYPE(NUNO).EQ.C)GC TC 335
     IF(TYPE(NUNC).EQ.CL)GC TC 337
     KLNC=0
     KONSO(LIST)=C
     IXPCN(LIST)=C
     GC TC 341
```

```
 333 IXPCN(LIST)=C                                                   SNA10800
     KCNSC(LIST)=1                                                   SNA1C810
     KLNC=1                                                          SNA10820
     GC TO 341                                                       SNA1C830
 335 IXPCN(LIST)=-1                                                  SNA1C840
     KCNSC(LIST)=1                                                   SNA1C850
     KLNC=1                                                          SNA1C860
     GC TO 341                                                       SNA1C870
 337 IXPCN(LIST)=1                                                   SNA1C880
     KCNSC(LIST)=C                                                   SNA1C890
     KLNC=C                                                          SNA1C900
 341 IF(IQLAL(NUNC).EQ.IC)GC TO 339                                  SNA1C910
     NEST(LIST)=1                                                    SNA1C920
     WEIGT(LIST)=1.                                                  SNA1C930
     GC TO 348                                                       SNA1C940
 339 IF(KLNC)904,904,906                                             SNA1C950
 904 WEIGT(LIST)=CVAL(NUNC)                                          SNA1C960
     GC TO 348                                                       SNA1C970
 906 WEIGT(LIST)=1./CVAL(NUNC)                                       SNA1C980
 348 CCNTINLE                                                        SNA1C990
 347 CCNTINLE                                                        SNA11000
     NLNC=NCBY                                                       SNA11C10
 361 LIST=LIST+1                                                     SNA11C20
     NFIRST(LIST)=NLNC                                               SNA11030
     NLAST(LIST)=KI                                                  SNA11C40
     SYMBUL(LIST)=SYM(KI)                                            SNA11C50
     IXPCN(LIST)=C                                                   SNA11C60
     IF(IQLAL(KI).EC.IQ)GC TC 371                                    SNA11C70
     NEST(LIST)=1                                                    SNA11080
     WEIGT(LIST)=1.                                                  SNA11090
     GC TO 303                                                       SNA11100
 371 WEIGT(LIST)=VAL(KI)                                             SNA11110
 303 CCNTINLE                                                        SNA11120
 305 CCNTINLE                                                        SNA11130
C                                                                    SNA11140
C       SUBPRCGRAM 'H'                                               SNA11150
C       THIS PRCGRAM SETS UP SFG INFC. FCR CV                        SNA11160
C       TYPE CONTROL SCURCES                                         SNA11170
 460 IF(LC)515,515,464                                               SNA11180
 464 DC 405 MI=1,LO                                                  SNA11190
     LI=ICV(MI)                                                      SNA11200
     NLNC=NLML(LI)                                                   SNA11210
     IF (LINC(NUNC)) 463,463,461                                     SNA11220
 463 LIST=LIST+1                                                     SNA11230
     NFIRST(LIST)=NLML(LI)                                           SNA11240
     NCBY=NCBY+1                                                     SNA11250
     NLAST(LIST)=NCBY                                                SNA11260
     SYMBUL(LIST)=SYM(NUNC)                                          SNA11270
     IF (TYPB(NUNC).EQ.Z) GC TC 433                                  SNA11280
     IF (TYPB(NUNC).EQ.R) GC TC 433                                  SNA11290
```

```
      IF (TYPB(NUNC).EQ.CL)GC TC 435
      IF (TYPB(NUNC).EQ.C) GC TC 437
      KLNC=C
      KCNSO(LIST)=C
      IXPCN(LIST)=C
      GC TC 441
  433 IXPCN(LIST)=C
      KCNSC(LIST)=1
      KLNC=1
      GC TC 441
  435 IXPCN(LIST)=-1
      KCNSC(LIST)=1
      KLNC=1
      GC TC 441
  437 IXPCN(LIST)=1
      KCNSC(LIST)=C
      KLNC=0
  441 IF (IQUAL(NUNO).EQ.IC) GC TC 439
      NEST(LIST)=1
      WEIGT(LIST)=1.
      GC TC 448
  439 IF (KUNO) 908,908,910
  908 WEIGT(LIST)=VAL(NUNC)
      GC TC 448
  910 WEIGT(LIST)=1./VAL(NUNC)
  448 CCNTINLE
  447 CCNTINLE
      NUNC=NCBY
  461 LIST=LIST+1
      NFIRST(LIST)=NUNO
      NLAST(LIST)=LI
      SYMBUL(LIST)=SYM(LI)
      IXPCN(LIST)=C
      IF (IQUAL(LI).EC.IC) GC TC 471
      NEST(LIST)=1
      WEIGT(LIST)=1.
      GC TC 403
  471 WEIGT(LIST)=VAL(LI)
  403 CCNTINLE
  405 CCNTINLE
C
C       SUBPRCGRAM 'I'
C       GENERATING CUTPUT LIST CF SFG
C
  515 CCNTINLE
      IF (NOUT) 514,512,514
  512 CALL TREP(NCCA,NODB,NF,NP,NPL)
      NCUT=NCBY+1
      MCPU=NPL-1
      DC 510 MOP=1,MCPU
```

```
      N1=NP(MOP)                                                        SNA11800
      N2=NP(MOP+1)                                                      SNA11810
      LIST=LIST+1                                                       SNA11820
      NFIRST(LIST)=IB(N1,N2)                                            SNA11830
      NLAST(LIST)=NCUT                                                  SNA11840
      SYMBUL(LIST)=ONE                                                  SNA11850
      IXPON(LIST)=0                                                     SNA11860
      KONSO(LIST)=0                                                     SNA11870
      NEST(LIST)=0                                                      SNA11880
  510 WEIGT(LIST)=NS(N1,N2)                                             SNA11890
  511 CONTINUE                                                          SNA11900
  514 NFIRST(1)=NCUT                                                    SNA11910
       NLAST(1)=NIN                                                     SNA11911
  482 IF (LISTG) 486,486,1200                                           SNA11920
 1200 WRITE (6,263)                                                     SNA11930
 263   FORMAT(1X,37HINITIAL TERMINAL EXPONENT  BRANCH                   SNA11940
     /35HBRANCH  1 IF SYMBOL      1 IF SYMBOL)                          SNA11950
      WRITE(6,264)                                                      SNA11960
  264 FORMAT (1X,33H NODE      NODE     OF S      VALUE,                SNA11970
     /37H     SYMBOL  IS INVERTED         IS USED)                      SNA11980
      DO 1202 J=1,LIST                                                  SNA11990
      WRITE (6,485) NFIRST(J),NLAST(J),IXPON(J),WEIGT(J),               SNA12000
     /SYMBUL(J),KONSO(J),NEST(J)                                        SNA12010
  485 FORMAT (3X,I2,7X,I2,6X,I2,4X,E12.5,1X,A3,8X,I2,14X,I2)            SNA12020
 1202 CONTINUE                                                          SNA12030
  486 CONTINUE                                                          SNA12040
C                                                                       SNA12050
C       SUBPROGRAM 'J'                                                  SNA12060
C       THIS PROGRAM ORDERS SFG INFORMATION                             SNA12070
C       FOR INPUT TO MAIN PROGRAM                                       SNA12080
      DO 87 J=1,NBG                                                     SNA12090
   87 MIX(J)=J                                                          SNA12100
      KONU=LIST-1                                                       SNA12110
      DO 80 KON=1,KONU                                                  SNA12120
      IU=KON+1                                                          SNA12130
      IL=KON                                                            SNA12140
      GO TO 83                                                          SNA12150
   81 MXL=MIX(IL)                                                       SNA12160
      MIX(IL)=MIX(IU)                                                   SNA12170
      MIX(IU)=MXL                                                       SNA12180
      IL=IL-1                                                           SNA12190
      IU=IU-1                                                           SNA12200
      IF (IL) 80,80,83                                                  SNA12210
   83 MIU=MIX(IU)                                                       SNA12220
      MIL=MIX(IL)                                                       SNA12230
      IF (NFIRST(MIU)-NFIRST(MIL)) 81,89,80                             SNA12240
   84 MXL=MIX(IL)                                                       SNA12250
      MIX(IL)=MIX(IU)                                                   SNA12260
      MIX(IU)=MXL                                                       SNA12270
      IL=IL-1                                                           SNA12280
```

```
      IL=IL-1                                                           SNA12290
      IF (IL) 80,80,82                                                  SNA12300
   82 MIU=MIX(IU)                                                       SNA12310
      MIL=MIX(IL)                                                       SNA12320
      IF (NFIRST(MIU)-NFIRST(MIL)) 80,89,80                             SNA12330
   89 IF (NLAST(MIL)-NLAST(MIL))80,80,84                                SNA12340
   80 CONTINUE                                                          SNA12350
 1305 CONTINUE                                                          SNA12360
      RETURN                                                            SNA12370
      END                                                               SNA12380
C                                                                       SNA12390
      SUBROUTINE FTREE(TYPX,JBX,LBX,INTRE,NOTREE,NOD,NOB)               SNA12400
C***********************************************************************SNA12410
C  THE FOLLOWING ARRAYS ARE ASSOCIATED WITH THE NETWORK                 SNA12420
C  CHARACTERISTICS NBN, AND NSPT (DEFINED IN PROGRAM MAIN-1)            SNA12430
      DIMENSION TYPX(25),JBX(25),LBX(25),INTRE(25),NOTREE(25)           SNA12440
      DIMENSION NP(25),NF(25,25),KCOL(16)                               SNA12450
C***********************************************************************SNA12460
      COMMON/C2/NNG,NBG                                                 SNA12470
      DATA E,VV,CV/2HE ,2HVV,2HCV/                                      SNA12480
      DATA R,CL,C,Y,Z/2HR ,2HL ,2HC ,2HY ,2HZ /                         SNA12490
      DATA G/2HG /                                                      SNA12500
      DO 40 I2=1,NNG                                                    SNA12510
      DO 40 I3=1,NNG                                                    SNA12520
   40 NF (I2,I3)=0                                                      SNA12530
      M=0                                                               SNA12540
      K=0                                                               SNA12550
      KC=0                                                              SNA12560
      DO 1 I=1,NOD                                                      SNA12570
    1 KCOL(I)=0                                                         SNA12580
      DO 3 I7=1,NOB                                                     SNA12590
    3 NOTREE(I7)=0                                                      SNA12600
      I=0                                                               SNA12610
    5 I=I+1                                                             SNA12620
      IF (TYPX(I).EQ.E) GO TO 10                                        SNA12630
    6 IF (TYPX(I).EQ.VV)GO TO 10                                        SNA12640
    8 IF (TYPX(I).EQ.CV)GO TO 10                                        SNA12650
      GO TO 4                                                           SNA12660
   10 K=K+1                                                             SNA12670
   14 INTRE(K)=I                                                        SNA12680
      JBX1=JBX(I)                                                       SNA12690
      KCOL(JBX1)=KCOL(JBX1)+1                                           SNA12700
      KCOL1=KCOL(JBX1)                                                  SNA12710
      NF(JBX1,KCOL1)=LBX(I)                                             SNA12720
      IBX1=LBX(I)                                                       SNA12730
      KCOL(IBX1)=KCOL(IBX1)+1                                           SNA12740
      KCOL2=KCOL(IBX1)                                                  SNA12750
      NF (IBX1,KCOL2)=JBX1                                              SNA12760
      NOTREE(I)=1                                                       SNA12770
      IF (K-NOD+1) 2,22,22                                              SNA12780
```

```
    2 IF (M) 4,4,12
    4 IF (I-NOB) 5,12,12
   12 M=M+1
      IF (TYPX(M).EQ.R) GO TO 16
      IF (TYPX(M).EQ.G) GO TO 16
   17 IF (TYPX(M).EQ.CL)GO TO 16
   18 IF (TYPX(M).EQ.C) GO TO 16
   19 IF (TYPX(M).EQ.Y) GO TO 16
   20 IF (TYPX(M).EQ.Z) GO TO 16
      IF (M-NOB) 12,22,22
   16 NINX=JBX(M)
      NCUTX=LBX(M)
      CALL TREP (NINX,NCUTX,NF,NP,NPL)
      IF (NPL) 21,21,12
   21 I=M
      GO TO 10
   22 CONTINUE
      RETURN
      END
C
      SUBROUTINE TREP (NIN,NCUT,NF,NP,NPL)
C*********************************************************************
C  THE FOLLOWING ARRAYS ARE ASSOCIATED WITH THE NETWORK
C  CHARACTERISTIC NBN(DEFINED IN PROGRAM MAIN-1)
      DIMENSION JX(25),NP(25),JMEM(25),KMEM(25)
      DIMENSION NF(25,25)
C*********************************************************************
      COMMON/C2/NNG,NBG
      DO 80 I5=1,NNG
      JX(I5)=0
      NP(I5)=0
      JMEM(I5)=0
   80 KMEM(I5)=0
      NPL=0
      JX(1)=NIN
      JX(2)=NIN
      I=1
      J=NIN
      NP(I)=NIN
   20 K=0
   25 K=K+1
      IF (NF(J,K)-NCUT) 30,50,30
   30 IF (NF(J,K)) 34,32,34
   32 IF (J-NIN) 60,100,60
C
C       FLOWER CHECK
   34 NJK=NF(J,K)
      IF (NJK-JX(I)) 45,25,45
C
C       STORE AND REMEMBER VERTEX
```

```
   45 I=I+1
      NF(I)=NF(J,K)
      JMEM(I)=J
      IA=I+1
      JX(IA)=NF(J,K)
   42 J=NF(J,K)
      KMEM(I)=K
      GO TO 20
C
C       BACKSTEP
   60 J=JMEM(I)
      K=KMEM(I)
      I=I-1
      GO TO 25
C
C       FINAL PATH VERTEX AND PATH LENGTH
   50 II=I+1
      NF(II)=NCUT
   62 NPL=II
  100 CONTINUE
      RETURN
      END
C
      SUBROUTINE ARRAY (JSIG,XCON,JXPO,JKOD,POLY,LIL,KIK)
C***********************************************************************
C  THE FOLLOWING ARRAYS ARE ASSOCIATED WITH THE NETWORK
C  CHARACTERISTICS NTC, AND NEXPS (DEFINED IN PROGRAM MAIN-1)
      DIMENSION MSORT(12),KSORT(125),POLY(12,125)
C***********************************************************************
      COMMON/C1/MSORT,KSORT
      COMMON/C3/NEXPS,NTC
      MMX=0
      NNX=0
      IF (KIK-1) 3,22,3
    3 MMU=KIK-1
      DO 2 MM=1,MMU
      MMX=MMX+1
      IF (JXPO-MSORT(MM)) 2,10,2
    2 CONTINUE
   22 MSORT(KIK)=JXPO
      MMX=KIK
      KIK=KIK+1
      IF (KIK-NEXPS-1) 1386,1385,1385
 1385 WRITE(6,1387)
 1387 FORMAT (1X,43H S-POWER EXCEEDS L+M+T-+NC-EASE D+MENS+ONS ,
     /16HCONTAINING NEXPS)
 1386 CONTINUE
   10 IF (LIL-1) 11,24,11
   11 NNU=LIL-1
      DO 12 NN=1,NNU
```

```
      NNX=NNX+1
      IF (JKOD-KSORT(NN)) 12,20,12
   12 CONTINUE
   24 KSORT(LIL)=JKOD
      NNX=LIL
      LIL=LIL+1
      IF (LIL-NTO-1) 1367,1365,1365
 1365 WRITE (6,1366)
 1366 FORMAT (1X,46HNO. OF TERMS IN OUTPUT EXCEEDS LIMIT-INCREASE ,
     /25HDIMENSIONS CONTAINING NTO)
 1367 CONTINUE
   20 POLY(MMX,NNX)=POLY(MMX,NNX)+XCON*(-1.)**JSIG
      RETURN
      END
C
      SUBROUTINE DECODE (KOD,KODY,IZ,FB,JZ,SEMBOL,KODF,KODI,ITOP,KBASIS)
C***********************************************************************
C  THE FOLLOWING ARRAYS ARE ASSOCIATED WITH THE NETWORK
C  CHARACTERISTICS NSPT, AND NTO (DEFINED IN PROGRAM MAIN -1)
      DIMENSION ITOP(125),SEMBOL(16),KODF(16),KODI(16)
C***********************************************************************
      COMMON/C4/NSPT
      IZ=0
      M=KBASIS-1
      DO 3 J=1,KOD
      CALL IAND (KODY,M,IPOWER,1,KBASIS)
      IF (IPOWER) 3,3,2
    2 IF (SEMBOL(J).EQ.FB) GO TO 4
      IZ=IZ+1
      IF (IZ-NSPT-1) 1371,1370,1370
 1370 WRITE (6,1372)
 1372 FORMAT (1X,48HNO. OF SYMBOLS PER TERM EXCEEDS OUTPUT-INCREASE ,
     /26HDIMENSIONS CONTAINING NSPT)
 1371 CONTINUE
      KODF(IZ)=IPOWER
      KODI(IZ)=J
      GO TO 3
    4 ITOP(JZ)=1
    3 KODY=KODY/KBASIS
      RETURN
      END
C
      SUBROUTINE IAND (MX,NX,MN,IFLAG,KBA)
      M=MX
      N=NX
      IF (IFLAG.EQ.0) GO TO 5
      KBASIS=KBA
      DO 6 K=1,64
      KBASIS=KBASIS/2
      IF (KBASIS-1) 6,8,6
```

```
      CONTINUE                                                          SNA14290
      LAST=K                                                            SNA14300
      GO TO 7                                                           SNA14310
5     LAST=25                                                           SNA14320
      25 IS THE MAXIMUM NO. OF NODES IN SFG. CHANGE AS NEEDED           SNA14330
      MN=0                                                              SNA14340
      NTHTWO=1                                                          SNA14350
      DO 10 I=1,LAST                                                    SNA14360
      NTHTWO=NTHTWO*2                                                   SNA14370
      NTEMP=N/2                                                         SNA14380
      NTEMP=NTEMP*2                                                     SNA14390
      IF(N-NTEMP)3,1,3                                                  SNA14400
      MTEMP=M/2                                                         SNA14410
      MTEMP=MTEMP*2                                                     SNA14420
      IF(M-MTEMP)2,1,2                                                  SNA14430
2     MN=MN+NTHTWO/2                                                    SNA14440
      IF(IFLAG)1,4,1                                                    SNA14450
      M=M/2                                                             SNA14460
      N=N/2                                                             SNA14470
0     CONTINUE                                                          SNA14480
      RETURN                                                            SNA14490
      END                                                               SNA14500
```

```
// FOR SNAP
*NON PROCESS PROGRAM
*IOCS(TYPEWRITER,CARD,PLOTTER)
*ONE WORD INTEGERS
C
C                            ****SNAP****
C         THIS PROGRAM FINDS THE SYMBOLIC TRANSFER
C         FUNCTION OR IMMITANCE FUNCTION OF A
C         LUMPED LINEAR TIME INVARIANT NETWORK.
C
      EXTERNAL PART3
      DIMENSION NS(15,15),NF(15,15),IB(15,15)
      DIMENSION TYPB(15),JB(15),LB(15),SYM(15),IQUAL(15),VAL(15)
      DIMENSION NUML(15),INTEE(15),LINC(15),IVV(15),ICV(15)
      DIMENSION NOTRE(15),INTRE(15),NUMX(15),TYPX(15),JBX(15),LBX(15)
      DIMENSION SYMX(15),IQUAX(15),VALX(15),NUMLX(15)
      DIMENSION KONS(8),KODI(8),SEMBL(8),KODF(8)
      DIMENSION NFIRS(30),NLAST(30),IXPON(30),WEIGT(30)
      DIMENSION MSORT(5),KSORT(40)
      DIMENSION SYMBU(30),KONSO(30),NEST(30)
      COMMON NBN,NBG,NTO,NSPT,NEXPS,NPAC,NRI,NEON,NRS
      COMMON NNG,NSPTU,NBTG
      COMMON NOD,NOB,KBASI,LISTG,LISTC,LISTP
      COMMON NIN,NOUT,NODA,NODB
      COMMON NFIRS,NLAST,IXPON,WEIGT,SYMBU,KONSO,NEST,LIST
      COMMON  KODI,KONS,KODF,SEMBL,MSORT,KSORT
      COMMON LIL,KIK,KUO,IZ
      COMMON NCI
      NOUT=0
      CALL SUB1(IB,NS,NF,TYPB,JB,LB,SYM,IQUAL,VAL,NUML,INTEE,LINC,KLU,
     1IVV,ICV,MNB,MO,LO,NOTRE,INTRE,NUMX,TYPX,JBX,LBX,SYMX,IQUAX,VALX,NU
     1MLX)
      IF(MNB)1,1,2
 2    STOP
 1    CONTINUE
      CALL SUB2(IB,NS,NF,TYPB,JB,LB,SYM,IQUAL,VAL,NUML,INTEE,LINC,KLU,
     1IVV,ICV,MO,LO,NOTRE,INTRE,NUMX,TYPX,JBX,LBX,SYMX,IQUAX,VALX,NUMLX)
      CALL LINK(PART3)
      END
// DUP
*STORECI             SNAP   SNAP
*LOCAL (SUB1,FREP,FTREE),(SUB2,TREP)
*CCEND



// FOR SUB1
      SUBROUTINE SUB1(IB,NS,NF,TYPB,JB,LB,SYM,IQUAL,VAL,NUML,INTEE,
     1LINC,KLU,IVV,ICV,MNB,MO,LO,NOTRE,INTRE,NUMX,TYPX,JBX,LBX,SYMX,IQUA
     1X,VALX,NUMLX)
      REAL IBLAN
      DIMENSION JROW(15)
      DIMENSION SN(6)
      DIMENSION NP(15)
```

```
      DIMENSION SM(9)
      DIMENSION NS(15,15),NF(15,15),IB(15,15)
       DIMENSION NUML(15),INTEE(15),LINC(15),IVV(15),ICV(15)
       DIMENSION TYPB(15),JB(15),LB(15),SYM(15),IQUAL(15),VAL(15)
       DIMENSION SYMX(15),IQUAX(15),VALX(15),NUMLX(15)
      DIMENSION NOTRE(15),INTRE(15),NUMX(15),TYPX(15),JBX(15),LBX(15)
       DIMENSION NFIRS(30),NLAST(30),IXPON(30),WEIGT(30)
       DIMENSION SYMBU(30),KONSO(30),NEST(30)
      DIMENSION KONS(8),KODI(8),SEMBL(8),KODF(8)
      DIMENSION MSORT(5),KSORT(40)
       COMMON NBN,NBG,NTO,NSPT,NEXPS,NPAC,NRI,NEON,NRS
       COMMON NNG,NSPTU,NBTG
       COMMON NOD,NOB,KBASI,LISTG,LISTC,LISTP
       COMMON NIN,NOUT,NODA,NODB
       COMMON NFIRS,NLAST,IXPON,WEIGT,SYMBU,KONSO,NEST,LIST
      COMMON KODI,KONS,KODF,SEMBL,MSORT,KSORT
      COMMON LIL,KIK,KOO,IZ
      COMMON NCI
      DATA SM(1),SM(2),SM(3)/'AAA','BBB','CCC'/
       DATA SM(4),SM(5),SM(6)/'DDD','EEE','FFF'/
       DATA SM(7),SM(8),SM(9)/'PPP','QQQ','RRR'/
      DATA SN(1),SN(2),SN(3)/'K1','K2','K3'/
      DATA SN(4),SN(5),SN(6)/'K4','K5','K6'/
      DATA FB/' FB'/
      DATA IBLAN/' '/
      DATA CC,CV,VV,VC/'CC','CV','VV','VC'/
C
C                     PRELIMINARY INPUT INFORMATION
C     NBN=NUMBER OF BRANCHES IN NETWORK.
C     NBG=NO. OF BRANCHES  OF SFG
C     NTO=NO. OF TERMS IN OUTPUT.
C     NSPT=NO . OF SYMBOLS PER TERM.
C     NEXPS=NO. OF DIFFERENT POWERS OF S
C     NPAC=NO. OF PATHS PLUS CIRCUITS.
C     NRI=MAXIMUM NUMBER OF NONTOUCHING LOOPS.
C     NCI=MAXIMUM NUMBER OF LOOPS NOT TOUCHING ANY GIVEN LOOP
C     NEON=NUMBER OF NONTOUCHING PAIRS OF LOOPS
C     NRS=NUMBER OF REPEATED SYMBOLS (NUMBER OF NETWORK
C     ELEMENTS ASSIGNED SAME SYMBOL)
       NBN=15
       NBG=30
       NTO=40
       NSPT=8
       NEXPS=5
       NPAC=125
      NRI=8
      NRS=9
       NCI=40
      NEON =400
       KJ=0
C     NSPTU=NUMBER OF SYMBOLS IN NUMERATOR OF EACH TERM
C     NBTG=NUMBER OF BRANCHES OF TREE OF SFG
C     NNG=NUMBER OF NODES IN SFG
      NNG=NBN
      NSPTU=NSPT/2
      NBTG=NBN
```

```
      WRITE(1,260)
C     THE NEXT 6 CARDS ARE FOR PROBLEM IDENTIFICATION ON THE 1ST DATA CARD
      READ (5,1150)(WEIGT(J),J=1,72)
 1150 FORMAT(72A1)
      IF(WEIGT(1)-IBLAN)9001,9000,9001
 9000  MNB=1
       GO TO 9998
 9001 CONTINUE
      WRITE (1,1160)(WEIGT(J),J=1,71)
 1160 FORMAT(1X,71A1//)
      DO 1151 J=1,72
 1151 WEIGT(J)=0.
      READ(5,1240)NOD,NOB,KBASI   ,LISTG,LISTC,LISTP
 1240 FORMAT(3I5,5X,3I1)
      IF(KBASI)1357,1357,1358
1357   KBASI=4
 1358 CONTINUE
       READ(5,1)NINN,NOOUT
  1   FORMAT(2I5)
      WRITE(1,720)NOD
720   FORMAT(2X,'NUMBER OF NODES=',I3)
        WRITE(1,721)NOB
  721 FORMAT(2X,'NUMBER OF BRANCHES=',I3)
      IF(LISTG)723,723,722
  722  CONTINUE
C     LIST SFG
  723  IF (LISTC)725,725,724
  724 CONTINUE
C     LIST ALL CIRCUITS
  725  IF(LISTP)726,726,727
  727   WRITE(6,728)
  728  FORMAT(2X,'LIST ALL PATHS FROM NODE',       I3,2X,'TO NODE ',I3)
726    WRITE(1,729)NINN
 729  FORMAT(2X,'NO. OF INPUT TERMINALS=',I3)
      WRITE(1,730)NOOUT
 730  FORMAT(2X,' NUMBER OF OUTPUT-TERMINALS = ',I3)
      WRITE(1,850)KBASI
  850  FORMAT(2X,'BASE FOR SYMBOL CODES=',I4)
C
C      SUBPROGRAM 'A'
       DO 152 IG=1,NBG
      NEST(IG)=0
  152 KONSO(IG)=0
      DO 710 IC=1,NNG
      DO 710 IK=1,NNG
      NS(IC,IK)=0
  710 NF(IC,IK)=0
      LIST=1
      MO=0
      LO=0
      IXPON(1)=0
       WEIGT(1)=-1.
       SYMBU(1)=FB
      KONSO(1)=0
      NEST(1)=1
      DO 5 I1=1,NNG
```

```
    5 JROW(I1)=0
      DO 528 I=1,NOB
      READ(5,9) TYPX(I),NUMX(I),JBX(I),LBX(I),SYMX(I),
     1IQUAX(I),VALX(I),NUMLX(I)
      IF(TYPX(I)-CC)9036,1300,9036
 9036 IF(TYPX(I)-CV)9037,1300,9037
 9037 IF(TYPX(I)-VV)9038,1300,9038
 9038 IF(TYPX(I)-VC)1301,1300,1301
 1300 IF(NUMLX(I))1301,1302,1301
 1302 WRITE(6,1303)
 1303 FORMAT(1X,'   ***ERROR***CONTROL SPECIFICATION FOR DEP. SOURCE
     1  MISSING')
      GO TO 7000
 1301 CONTINUE
  528 CONTINUE
      GO TO 7777
 7000 NOB=0
       MNB=1
       GO TO 9998
    9 FORMAT (A2,I3,2I5, 1X,A3,A1,E12.5,I3)
 7777 CONTINUE
       KJ=0
      MMM=1
      IF(NINN-1)222,222,333
 333   CONTINUE
       KJ=1
 222  READ(5,224)NIN,K
 224   FORMAT(2I5)
      WRITE(1,225)NIN
 225   FORMAT(1X,'ELEMENT NO. OF SOURCE ='I3)
      IF(KJ)936,936,937
 937   N=NINN-1
      DO 226 I=1,N
      READ(5,227)NI,M
 227   FORMAT(2I5)
       WRITE(1,928)I,NI
 928  FORMAT(1X,'ELEMENT NO. OF SOURCE (',I2,')=',I3)
      IF(K)929,929,930
929   IF(M)931,931,932
 931  TYPX(NI)=VV
      GO TO 935
 932   TYPX(NI)=VC
      GO TO 935
 930  IF(M)933,933,934
 933   TYPX(NI)=CV
      GO TO 935
 934  TYPX(NI)=CC
 935   SYMX(NI)=SN(MMM)
      NUMLX(NI)=NIN
      MMM=MMM+1
226    CONTINUE
 936  CONTINUE
       KKK=1
 5559 CONTINUE
      WRITE(1,260)
       IF(KKK-1)2603,2602,2603
```

```
2602    WRITE(1,2600)
 2600   FORMAT(30X,'NETWORK')
        GO TO 2604
 2603   WRITE(1,2601)
2601    FORMAT(30X,'MODIFIED NETWORK')
 2604 CONTINUE
      WRITE(1,261)
 261  FORMAT(1X,'ELEMENT ELEMENT INTIAL TERMINAL ELEMENT ELEMENT ELEMENT
     1 NO.')
      WRITE(1,262)
 262  FORMAT(1X,' TYPE   NUMBER   NODE     NODE   SYMBOL   VALUE  OF CON
     1TROL')
      DO 601 M=1,NOB
  601 WRITE(1,600) TYPX(M),NUMX(M),JBX(M),LBX(M),SYMX(M),
     1IQUAX(M),VALX(M),NUMLX(M)
 600  FORMAT(4X,A2,6X,I2,6X,I2,6X,I2,6X,A3,A1,E12.5,2X,I2)
       CALL FTREE(TYPX,JBX,LBX,INTRE,NOTRE)
      KLU=0
       DO 555 I1=1,NNG
      INTEE(I1)=0
 555   JROW(I1)=0
C
C      SUBPROGRAM 'B'
      WRITE(1,518)
  518 FORMAT(30X,13HTREE SELECTED)
      NUMU=NOD-1
      DO 21 NU=1,NUMU
      IO=INTRE(NU)
      NUMC=NUMX(IO)
      TYPB(NUMC)=TYPX(IO)
      JB(NUMC)=JBX(IO)
      LB(NUMC)=LBX(IO)
      SYM(NUMC)=SYMX(IO)
      IQUAL(NUMC)=IQUAX(IO)
      VAL(NUMC)=VALX(IO)
      NUML(NUMC)=NUMLX(IO)
       INTEE(NUMC)=1
      WRITE(1,517)TYPB(NUMC),NUMC,JB(NUMC),LB(NUMC),SYM(NUMC),
     1IQUAL(NUMC),VAL(NUMC),NUML(NUMC)
  517 FORMAT (4X,A2,6X,I2,6X,I2,6X,I2,6X,A3,A1,E12.5,2X,I2)
      KLU=KLU+1
      LINC(NUMC)=0
      IF(TYPB(NUMC)-VV)3,9039,3
 9039  MO=MO+1
      IVV(MO)=NUMC
3     IF(TYPB(NUMC)-CV)4,9040,4
 9040 LO=LO+1
      ICV(LO)=NUMC
    4 JF=JB(NUMC)
      LF=LB(NUMC)
      IB(JF,LF)=NUMC
      IB(LF,JF)=NUMC
      JROW(JF)=JROW(JF)+1
      JROJ=JROW(JF)
      NF(JF,JROJ)=LF
      NS(JF,LF)=1
```

```
      JROW(LF)=JROW(LF)+1
       JROL=JROW(LF)
       NF(LF,JROL)=JF
      NS(LF,JF)=-1
   21 CONTINUE
       IF(KKK-1)6660,6661,6660
 6661  CONTINUE
       IF(NOOUT-1)8000,211,8000
8000   NODA=NOD
       MM=1
       WRITE(1,260)
 22   READ(5,12)NOUUT,NODAA,NODBB,K
       WRITE(1,260)
       IF(NOUUT)5561,5560,5561
5561   WRITE(1,5562)KJ,NOUUT
 5562  FORMAT(1X,'ELEMENT NUMBER ASSOCIATED WITH OUTPUT(',I1,')=',I3)
       GO TO 5565
5560   WRITE(1,5563)KJ,NODAA
 5563  FORMAT(1X,'POSITIVE OUTPUT VOLTAGE TERMINAL(',I1,')='I3)
       WRITE(1,5564)KJ,NODBB
5564   FORMAT(1X,'NEGATIVE OUTPUT VOLTAGE TERMINAL(',I1,')=',I3)
 5565  CONTINUE
       KJ=KJ+1
      IF(NOUUT)113,113,14
14    NOB=NOB+1
      IF(K) 15,15,16
 15   TYPX(NOB)=VV
      GO TO1117
 16   TYPX(NOB)=CV
1117   NUMX(NOB)=NOB
       JBX(NOB)=NOD
      LBX(NOB)=NOD+1
      SYMX(NOB)=SM(MM)
      MM=MM+1
        NUMLX(NOB)=NOUUT
       NOD=NOD+1
       GO TO 17
113   CALL FREP (NODAA,NODBB,NF,NP,NPL)
      NPLL=NPL-1
      DO 18 I=1,NPLL
       NOB=NOB+1
       TYPX(NOB)=VV
      NUMX(NOB)=NOB
       JBX(NOB)=NOD
       LBX(NOB)=NOD+1
      SYMX(NOB)=SM(MM)
       NP1=NP(I)
       NP2=NP(I+1)
      NUMLX(NOB)=IB(NP1,NP2)
       NOD=NOD+1
 18   CONTINUE
        MM=MM+1
 17   NODB=NOD
       IF(NOOUT-1)20,20,8001
 8001 NOOUT=NOOUT-1
       GO TO 22
```

```
 211   READ(5,12) NOUT,NODA,NODB,K
        WRITE(1,260)
        IF(NOUT)5550,5551,5550
 5550   WRITE(1,5555)NOUT
 5555   FORMAT(1X,'ELEMENT NUMBER ASSOCIATED WITH OUTPUT=',I3)
        GO TO 6660
 5551  WRITE(1,5556)NODA
 5556   FORMAT(1X,'POSITIVE  OUTPUT VOLTAGE TERMINAL=',I3)
        WRITE(1,5557)NODB
 5557   FORMAT(1X,'NEGATIVE  OUTPUT  VOLTAGE  TERMINAL=',I3)
        GO TO 6660
 20     KKK=KKK-1
        GO TO 5559
 12     FORMAT(4I5)
 6660  DO 13 ILL=1,NOD
       JROI=JROW(ILL)+1
    13 NF(ILL,JROI)=0
       WRITE(1,260)
 260   FORMAT(//)
       WRITE(1,715)
 715    FORMAT(30X,' SFG ',/)
 9998  RETURN
        END




// FOR SUB2
        SUBROUTINE SUB2(IB,NS,NF,TYPB,JB,LB,SYM,IQUAL,VAL,NUML,INTEE,
     1LINC,KLU,IVV,ICV,MC,LO,NOTRE,INTRE,NUMX,TYPX,JBX,LBX,SYMX,IQUAX,VA
     1LX,NUMLX)
        DIMENSION INTRE(15),JB(15),LB(15)
       DIMENSION TYPB(15),TYPX(15),JBX(15),LBX(15),SYMX(15)
       DIMENSION CVAL(30),MAPY(30),TYPE(30)
       DIMENSION I)UAX(15),INTEE(15),LINC(15),NP(15)
       DIMENSION NS(15,15),NF(15,15),IB(15,15)
       DIMENSION KONS(8),KODI(8),SEMBL(8),KODF(8)
        DIMENSION NFIRS(30),NLAST(30),IXPON(30),WEIGT(30)
       DIMENSION MSORT(5),KSORT(40)
        DIMENSION SYMBU(30),KONSO(30),NEST(30)
        DIMENSION VALX(15),NOTRE(15),SYM(15),IQUAL(15),VAL(15)
       DIMENSION NUMX(15),IVV(15),ICV(15),NUML(15),NUMLX(15)
        COMMON NBN,NBG,NTO,NSPT,NEXPS,NPAC,NRI,NEON,NRS
        COMMON NNG,NSPTU,NBTG
        COMMON NOD,NOB,KBASI,LISTG,LISTC,LISTP
        COMMON NIN,NOUT,NODA,NODB
        COMMON NFIRS,NLAST,IXPON,WEIGT,SYMBU,KONSO,NEST,LIST
        COMMON KODI,KONS,KODF,SEMBL,MSORT,KSORT
       COMMON LIL,KIK,KOO,IZ
        COMMON NCI
       DATA E,CI,CC,CV,VV,VC/'E ','I ','CC','CV','VV','VC'/
       DATA Y,G,C,IQ,R,CL,Z/'Y ','G ','C ','=','R ','L ','Z '/
        DATA ONE/'  1'/
        LINK=0
C
```

```
C       SUBPROGRAM 'C'
C       THIS PROGRAM GENERATES SIGNAL FLOW GRAPH INFO.
C       FROM BRANCH NODE TO LINK NODE
      NOBY=NOB
  151 CONTINUE
      NES=0
      LON=0
      IF(KLU-NOB)532,360,532
  532 LINK=LINK+1
      IF(NOTRE(LINK))534,534,532
  534 NUMC=NUMX(LINK)
      TYPE(NUMC)=TYPX(LINK)
       JK=JBX(LINK)
      LK=LBX(LINK)
       SYM(NUMC)=SYMX(LINK)
       IQUAL(NUMC)=IQUAX(LINK)
        NUMB=NUMLX(LINK)
       CVAL(NUMC)=VALX(LINK)
      TYP2=TYPE(NUMC)
      CVALU=CVAL(NUMC)
      KLU=KLU+1
      LINC(NUMC)=1
      KDEPS=0
      KANSO=0
      IF(TYPE(NUMC)-CL)9041,117,9041
 9041 IF(TYPE(NUMC)-G)9042,119,9042
 9042 IF(TYPE(NUMC)-Y)9043,119,9043
 9043 IF(TYPE(NUMC)-R)9044,700,9044
 9044 IF(TYPE(NUMC)-Z)9045,700,9045
 9045 IF(TYPE(NUMC)-C)9046,121,9046
 9046 CONTINUE
      KDEPS=1
       IF(TYPE(NUMC)-E)9047,123,9047
 9047 IF(TYPE(NUMC)-CI)9048,123,9048
 9048  IF(TYPE(NUMC)-VC)9049,165,9049
 9049  IF(TYPE(NUMC)-CC)117,265,117
  117 IXPS=-1
      KANSO=1
      GO TO 123
  119 IXPS=0
      GO TO 123
  700 IXPS=0
      KANSO=1
      GO TO 123
  121 IXPS=1
  123 CALL TREP(JK,LK,NF,NP,NPL)
      IFIN=NUMC
  149 LON=LON+1
      NP1=NP(LON)
      NP2=NP(LON+1)
  /07 INIT=IB(NP1.NP2)
  109 SIGH=NS(NP1,NP2)
      IF(KDEPS)167,167,169
 167  IF(IQUAL(NUMC)-IQ)9002,111,9002
9002   CONTINUE
       NES=1
```

```
      CONST=SIGH
      GO TO 125
  111 CONST=SIGH*CVALU
  125 LIST=LIST+1
      IF(NES)502,503,502
  502 NEST(LIST)=1
  503 KONSO(LIST)=KANSO
      NFIRS(LIST)=INIT
      NLAST(LIST)=IFIN
      SYMBU(LIST)=SYM(IFIN)
      IXPON(LIST)=IXPS
      IF(KONSO(LIST))505,505,504
  504 WEIGT(LIST)=1./CONST
      GO TO 506
  505 WEIGT(LIST).CONST
  506 MAPY(NUMC)=LIST
  127 FORMAT(3I5,E12.5)
  129 FORMAT (A4)
C
C      SUBPROGRAM 'D'
C      THIS PROGRAM GENERATES SIGNAL FLOW GRAPH INFO.
C      FROM LINK NODE TO BRANCH NODE
  169 CONTINUE
      IF(TYPB(INIT)-E)9050,201,9050
 9050  IF(TYPB(INIT)-CI)9051,201,9051
 9051 IF(TYPB(INIT)-VV)9052,201,9052
 9052 IF(TYPB(INIT)-CV)9053,201,9053
 9053  CONTINUE
       LIST=LIST+1
      IF(TYPB(INIT)-R)9054,133,9054
 9054 IF(TYPB(INIT)-Z)9055,133,9054
 9055 IF(TYPB(INIT)-G)9056,702,9056
 9056  IF(TYPB(INIT)-Y)9057,702,9057
 9057  IF(TYPB(INIT)-CL)9058,135,9058
 9058 IF(TYPB(INIT)-C)133,137,133
  133 IXPON(LIST)=0
      GO TO 141
  702 IXPON(LIST)=0
      KONSO(LIST)=1
      GO TO 141
  135 IXPON(LIST)=1
      GO TO 141
  137 IXPON(LIST)=-1
      KONSO(LIST)=1
 141  IF(IQUAL(INIT)-IQ)9999,139,9999
 9999 CONTINUE
      NEST(LIST)=1
      WEIGT(LIST)=-1.*SIGH
      GO TO 147
  139 IF(KONSO(LIST))608,608,607
  607 WEIGT(LIST)=-SIGH/VAL(INIT)
      GO TO 147
  608 WEIGT(LIST)=-SIGH*VAL(INIT)
 147  NFIRS(LIST)=IFIN
      NLAST(LIST)=INIT
       SYMBU(LIST)=SYM(INIT)
```

```
  201 NPLA=NPL-1-LON
      IF(NPLA)151,151,149
C
C      SUBPROGRAM 'E'
C      THIS PROGRAM SETS JP SFG INFO. FOR VC
C      TYPE CONTROL SOURCES
  165 NUNO=NUMB
       IF(INTEE(NUMB))163,163,161
  163 LIST=LIST+1
      NFIRS(LIST)=NJMB
      NOBY=NOBY+1
      SYMBU(LIST)=SYM(NUMB)
      NLAST(LIST)=NOBY
      NUNO=NOBY
      IF(TYPE(NUMB)-Y)9059,912,9059
 9059 IF(TYPE(NUMB)-G)9060,912,9060
 9060 IF(TYPE(NUMB)-C)9061,914,9061
 9061 IF(TYPE(NUMB)-CL)9062,916,9062
 9062  CONTINUE
      KUNO=0
      IXPON(LIST)=0
      GO TO 918
  912 IXPON(LIST)=0
      KUNO=1
      GO TO 918
  914 IXPON(LIST)=-1
      KUNO=1
      GO TO 918
  916 IXPON(LIST)=1
      KUNO=0
 918  IF(IQUAL(NUMB)-IQ)9063,920,9063
 9063  CONTINUE
      NEST(LIST)=1
      WEIGT(LIST)=1.
      GO TO 209
  920 IF(KUNO)922,922,924
  922 WEIGT(LIST)=CVAL(NUMB)
      GO TO 209
  924 WEIGT(LIST)=1./CVAL(NUMB)
209    KONSO(LIST)=KUNO
  161 LIST=LIST+1
      NLAST(LIST)=NJMC
       NFIRS(LIST)=NUNO
      SYMBU(LIST)=SYM(NUMC)
      IXPON(LIST)=0
       IF(IQUAL(NUMC)-IQ)9064,171,9064
 9064 CONTINUE
      NEST(LIST)=1
      WEIGT(LIST)=1.
       GO TO 203
  171 WEIGT(LIST)=CVALU
  203 CONTINUE
      GO TO 123
C
C      SUBPROGRAM 'F'
C      THIS PROGRAM SETS UP SFG INFO. FOR CC
```

```
C        TYPE CONTROL SOURCES
  265 MUNO=NUMB
      IF(INTEE(NUMB))621,621,620
  620 LIST=LIST+1
      NFIRS(LIST)=NUMB
      NOBY=NOBY+1
      NLAST(LIST)=NOBY
      SYMBU(LIST)=SYM(NUMB)
      MUNO=NOBY
       IF(TYPB(NUMB)-Z)9065,233,9065
 9065 IF(TYPB(NUMB)-R)9066,233,9066
 9066 IF(TYPB(NUMB)-CL)9067,235,9067
 9067 IF(TYPB(NUMB)-C)9068,237,9068
 9068 CONTINUE
      KUNO=0
      IXPON(LIST)=0
       GO TO 241
  233 IXPON(LIST)=0
       KUNO=1
       GO TO 241
  235 IXPON(LIST)=-1
      KUNO=1
       GO TO 241
 237   IXPON(LIST)=1
      KUNO=0
 241  IF(IQUAL(NUMB)-IQ)9069,239,9069
 9069 CONTINUE
      NEST(LIST)=1
      WEIGT(LIST)=1
      GO TO 247
  239 IF(KUNO)900,900,902
  900 WEIGT(LIST)=VAL(NUMB)
      GO TO 247
  902 WEIGT(LIST)=1./VAL(NUMB)
 247  KONSO(LIST)=1
  621 LIST=LIST+1
      NFIRS(LIST)=MUNO
      NLAST(LIST)=NUMC
      IXPON(LIST)=0
       SYMBU(LIST)=SYM(NUMC)
       IF(IQUAL(NUMC)-IQ)9029,271,9029
 9029 CONTINUE
       NEST(LIST)=1
       WEIGT(LIST)=1.
      GO TO 281
  271 WEIGT(LIST)=CVALU
  281 CONTINUE
      GO TO 123
C
C      SUBPROGRAM 'G'
C      THIS PROGRAM SETS UP SFG INFO. FOR VV
C      TYPE CONTROL SOURCES
  360 IF(MO)460,460,364
  364 DO 305 MI=1,MO
      KI=IVV(MI)
      NUNO=NUML(KI)
```

```
      IF(LINC(NUNO))361,361,363
  363 LIST=LIST+1
       NFIRS(LIST)=NUML(KI)
      NOBY=NOBY+1
      NLAST(LIST)=NOBY
       SYMBU(LIST)=SYM(NUNO)
      IF(TYPE(NUNO)-Y)9070,333,9070
 9070  IF(TYPE(NUNO)-G)9071,333,9071
 9071 IF(TYPE(NUNO)-C)9072,335,9072
 9072 IF(TYPE(NUNO)-CL)9073,337,9073
 9073 CONTINUE
      KUNO=0
      IXPON(LIST)=0
      GO TO 341
 333   IXPON(LIST)=0
      KUNO=1
      GO TO 341
  335 IXPON(LIST)=-1
      KUNO=1
      GO TO 341
  337 IXPON(LIST)=1
      KUNO=0
 341  IF(IQUAL(NUNO)-IQ)9074,339,9074
 9074 CONTINUE
      NEST(LIST)=1
      WEIGT(LIST)=1.
      GO TO 348
  339 IF(KUNO)904,904,906
  904 WEIGT(LIST)=CVAL(NUNO)
      GO TO 348
  906 WEIGT(LIST)=1./CVAL(NUNO)
  348 CONTINUE
  347 CONTINUE
      NUNO=NOBY
  361 LIST=LIST+1
       NFIRS(LIST)=NUNO
      NLAST(LIST)=KI
       SYMBU(LIST)=SYM(KI)
      IXPON(LIST)=0
       IF(IQUAL(KI)-IQ)9075,371,9075
 9075 CONTINUE
      NEST(LIST)=1
      WEIGT(LIST)=1.
      GO TO 303
  371 WEIGT(LIST)=VAL(KI)
  303 CONTINUE
  305 CONTINUE
C
C      SUBPROGRAM 'H'
C      THIS PROGRAM SETS UP SFG INFO. FOR CV
C      TYPE CONTROL SOURCES
  460 IF(LO)515,515,464
  464 DO 405 MI=1,LO
      LI=ICV(MI)
      NUNO=NUML(LI)
      IF (LINC(NUNO)) 463,463,461
```

```
  463 LIST=LIST+1
       NFIRS(LIST)=NUML(LI)
      NOBY=NOBY+1
      NLAST(LIST)=NOBY
        SYMBU(LIST)=SYM(NUNO)
       IF(TYPB(NUNO)-Z)9008,433,9008
9008   IF(TYPB(NUNO)-R)9009,433,9009
 9009  IF(TYPB(NUNO)-CL)9010,435,9010
 9010 IF(TYPB(NUNF)-C)9011,437,9011
 9011 CONTINUE
      KUNO=0
      IXPON(LIST)=0
      GO TO 441
  433 IXPON(LIST)=0
      KUNO=1
      GO TO 441
  735 IXPON(LIST)$-1
      KUNO=1
      GO TO 441
  437 IXPON(LIST)=1
      KUNO=0
 441  IF(IQUAL(NUNO)-IQ)9076,439,9076
 9076 CONTINUE
      NEST(LIST)=1
      WEIGT(LIST)=1.
      GO TO 448
  439 IF (KUNO) 908,908,910
  908 WEIGT(LIST)=VAL(NUNO)
      GO TO 448
  910 WEIGT(LIST)=1./VAL(NUNO)
 448  KONSO(LIST)=1
  447 CONTINUE
      NUNO=NOBY
  461 LIST=LIST+1
       NFIRS(LIST)=NUNO
      NLAST(LIST)=LI
       SYMBU(LIST)=SYM(LI)
      IXPON(LIST)=0
      IF(IQUAL(LI)-IQ)9077,471,9077
 9077 CONTINUE
      NEST(LIST)=1
      WEIGT(LIST)=1.
      GO TO 403
  471 WEIGT(LIST)=VAL(LI)
  403 CONTINUE
  405 CONTINUE
C
C      SUBPROGRAM 'I'
C      GENERATING OUTPUT LIST OF SFG
C
  515 CONTINUE
      IF (NOUT) 514,512,514
  512 CALL TREP(NODA,NODB,NF,NP,NPL)
      NOUT=NOBY+1
      MOPU=NPL-1
      DO 510 MOP=1,MOPU
```

```
      N1=NP(MOP)
      N2=NP(MOP+1)
      LIST=LIST+1
       NFIRS(LIST)=IB(N1,N2)
      NLAST(LIST)=NOUT
      SYMBU(LIST)=ONE
      IXPON(LIST)=0
      KONSO(LIST)=0
      NEST(LIST)=0
  510 WEIGT(LIST)=NS(N1,N2)
  511 CONTINUE
 514  NFIRS(1)=NOUT
      NLAST(1)=NIN
  482 IF (LISTG) 486,486,1200
 1200 WRITE (1,263)
  263 FORMAT (1X,'  INITIAL TERMINAL EXPONENT  BRANCH   BRANCH  1 IF SYM
     1BOL       1 IF SYMBOL')
      WRITE(1,264)
  264 FORMAT (1X,'    NODE      NODE     OF S     VALUE    VALUE  SYMBOL IS
     1 INVERTED IS USED')
      DO 1202 J=1,LIST
      WRITE(1,485) NFIRS(J),NLAST(J),IXPON(J),WEIGT(J),
     1SYMBU(J),KONSO(J),NEST(J)
 485  FORMAT(3X,I2,7X,I2,6X,I2,4X,E12.5,1X,A3,8X,I2,14X,I2)
 1202 CONTINUE
C
  486 CONTINUE
      WRITE(1,260)
260   FORMAT(//)
      RETURN
       END
// FOR FTREE
       SUBROUTINE FTREE(TYPX,JBX,LBX,INTRE,NOTRE)
C*********************************************************************
C  THE FOLLOWING ARRAYS ARE ASSOCIATED WITH THE NETWORK
C  CHARACTERISTICS NBN, AND NSPT
C*********************************************************************
       DIMENSION TYPX(15),JBX(15),LBX(15),INTRE(15),NOTRE(15)
      DIMENSION NP(15),NF(15,15),KCOL(8)
      DIMENSION KONS(8),KODI(8),SEMBL(8),KODF(8)
       DIMENSION NFIRS(30),NLAST(30),IXPON(30),WEIGT(30)
      DIMENSION MSORT(5),KSORT(40)
       DIMENSION SYMBU(30),KONSO(30),NEST(30)
       COMMON NBN,NBG,NTO,NSPT,NEXPS,NPAC,NRI,NEON,NRS
       COMMON NNG,NSPTU,NBTG
       COMMON NOD,NOB,KBASI,LISTG,LISTC,LISTP
       COMMON NIN,NOUT,NODA,NODB
       COMMON NFIRS,NLAST,IXPON,WEIGT,SYMBU,KONSO,NEST,LIST
      COMMON KODI,KONS,KODF,SEMBL,MSORT,KSORT
      COMMON LIL,KIK,KOO,IZ
      COMMON NCI
      DATA E,VV,CV/'E ','VV','CV'/
      DATA R,CL,C,Y,Z/'R ','L ','C ','Y ','Z '/
      DATA G/'G '/
      DO 40 I2=1,NNG
      DO 40 I3=1,NNG
```

```
   40 NF (I2,I3)=0
      M=0
      K=0
      KC=0
      DO 1 I=1,NOD
    1 KCOL(I)=0
      DO 3 I7=1,NOB
 3    NOTRE(I7)=0
      I=0
    5 I=I+1
      IF(TYPX(I)-E)6,10,6
 6    IF(TYPX(I)-VV)8,10,8
 8    IF(TYPX(I)-CV)4,10,4
   10 K=K+1
   14 INTRE(K)=I
      JBX1=JBX(I)
      KCOL(JBX1)=KCOL(JBX1)+1
      KCOL1=KCOL(JBX1)
      NF(JBX1,KCOL1)=LBX(I)
      IBX1=LBX(I)
      KCOL(IBX1)=KCOL(IBX1)+1
      KCOL2=KCOL(IBX1)
      NF (IBX1,KCOL2)=JBX1
      NOTRE(I)=1
      IF (K-NOD+1) 2,22,22
    2 IF (M) 4,4,12
    4 IF (I-NOB) 5,12,12
   12 M=M+1
      IF(TYPX(M)-R)9078,16,9078
 9078 IF(TYPX(M)-G)17,16,17
 17    IF(TYPX(M)-CL)18,16,18
 18    IF(TYPX(M)-C)19,16,19
 19    IF(TYPX(M)-Y)20,16,20
 20   IF(TYPX(M)-Z)9079,16,9079
 9079  CONTINUE
      IF (M-NOB) 12,22,22
   16 NINX=JBX(M)
      NOUTX=LBX(M)
      CALL FREP (NINX,NOUTX,NF,NP,NPL)
      IF (NPL) 21,21,12
   21 I=M
        GOTO10
   22 CONTINUE
      RETURN
      END


// FOR FREP
        SUBROUTINE FREP(NIK,NOUK,NF,NP,NPL)
C***********************************************************************
C  THE FOLLOWING ARRAYS ARE ASSOCIATED WITH THE NETWORK
C  CHARACTERISTIC NBN
C***********************************************************************
```

```
      DIMENSION JX(15),NP(15),JMEM(15),KMEM(15),NF(15,15)
      DIMENSION KONS(8),KODI(8),SEMBL(8),KODF(8)
      DIMENSION MSORT(5),KSORT(40)
      DIMENSION SYMBU(30),KONSO(30),NEST(30)
      DIMENSION NFIRS(30),NLAST(30),IXPON(30),WEIGT(30)
      COMMON NBN,NBG,NTO,NSPT,NEXPS,NPAC,NRI,NEON,NRS
      COMMON NNG,NSPTU,NBTG
      COMMON NOD,NOB,KBASI,LISTG,LISTC,LISTP
      COMMON NIN,NOUT,NODA,NODB
      COMMON NFIRS,NLAST,IXPON,WEIGT,SYMBU,KONSO,NEST,LIST
      COMMON KODI,KONS,KODF,SEMBL,MSORT,KSORT
      COMMON LIL,KIK,KOO,IZ
      COMMON NCI
      DO 80 I5=1,NNG
      JX(I5)=0
      NP(I5)=0
      JMEM(I5)=0
   80 KMEM(I5)=0
      NPL=0
      JX(1)=NIK
      JX(2)=NIK
      I=1
      J=NIK
      NP(1)=NIK
   20 K=0
   25 K=K+1
      IF(NF(J,K)-NOUK)30,50,30
   30 IF (NF(J,K)) 34,32,34
 32   IF(J-NIK)60,100,60
C
C      FLOWER CHECK
   34 NJK=NF(J,K)
      IF (NJK-JX(I)) 45,25,45
C
C      STORE AND REMEMBER VERTEX
   45 I=I+1
      NP(I)=NF(J,K)
      JMEM(I)=J
      IA=I+1
      JX(IA)=NF(J,K)
   42 J=NF(J,K)
      KMEM(I)=K
      GO TO 20
C
C      BACKSTEP
   60 J=JMEM(I)
      K=KMEM(I)
      I=I-1
      GO TO 25
C
C      FINAL PATH VERTEX AND PATH LENGTH
   50 II=I+1
      NP(II)=NOUK
   62 NPL=II
  100 CONTINUE
      RETURN
```

```
      END



// FOR TREP
      SUBROUTINE TREP(NIK,NOJK,NF,NP,NPL)
C*********************************************************************
C  THE FOLLOWING ARRAYS ARE ASSOCIATED WITH THE NETWORK
C  CHARACTERISTIC NBN
C*********************************************************************
      DIMENSION JX(15),NP(15),JMEM(15),KMEM(15),NF(15,15)
      DIMENSION KONS(8),KODI(8),SEMBL(8),KODF(8)
      DIMENSION MSORT(5),KSORT(40)
      DIMENSION SYMBU(30),KONSO(30),NEST(30)
      DIMENSION NFIRS(30),NLAST(30),IXPON(30),WEIGT(30)
      COMMON NBN,NBG,NTO,NSPT,NEXPS,NPAC,NRI,NEON,NRS
      COMMON NNG,NSPTU,NBTG
      COMMON NOD,NOB,KBASI,LISTG,LISTC,LISTP
      COMMON NIN,NOUT,NODA,NODB
      COMMON NFIRS,NLAST,IXPON,WEIGT,SYMBU,KONSO,NEST,LIST
      COMMON KODI,KONS,KODF,SEMBL,MSORT,KSORT
      COMMON LIL,KIK,KOO,IZ
      COMMON NCI
      DO 80 I5=1,NNG
      JX(I5)=0
      NP(I5)=0
      JMEM(I5)=0
   80 KMEM(I5)=0
      NPL=0
      JX(1)=NIK
      JX(2)=NIK
      I=1
      J=NIK
      NP(1)=NIK
   20 K=0
   25 K=K+1
      IF(NF(J,K)-NOJK)30,50,30
   30 IF (NF(J,K)) 34,32,34
 32   IF(J-NIK)60,100,60
C
C       FLOWER CHECK
   34 NJK=NF(J,K)
      IF (NJK-JX(I)) 45,25,45
C
C       STORE AND REMEMBER VERTEX
   45 I=I+1
      NP(I)=NF(J,K)
      JMEM(I)=J
      IA=I+1
      JX(IA)=NF(J,K)
   42 J=NF(J,K)
      KMEM(I)=K
      GO TO 20
C
```

```
C        BACKSTEP
   60 J=JMEM(I)
      K=KMEM(I)
      I=I-1
      GO TO 25
C
C        FINAL PATH VERTEX AND PATH LENGTH
   50 II=I+1
      NP(II)=NOUK
   62 NPL=II
  100 CONTINUE
      RETURN
      END
```

```
// FOR PART3
*IOCS(CARD,TYPEWRITER,PLOTTER)
*NON PROCESS PROGRAM
*ONE WORD INTEGERS
      INTEGER F
      DIMENSION SMBOL(30)
      DIMENSION N(15,15)
      DIMENSION KODE(15,15),IXPO(15,15),CONS(15,15)
      DIMENSION SIMBN(40,4),SIMBD(40,4)
      DIMENSION POLYU(5,40)
      DIMENSION AMAG(10,10),AARG(10,10)
      DIMENSION ISET(8,40)
      DIMENSION NA(40),NB(40)
      DIMENSION POLY(5,40),ITOP(40)
      DIMENSION KEP(40,4),KED(40,4)
      DIMENSION NPCOD(125),IXPOT(125),CONST(125),KODET(125)
      DIMENSION MIX(30)
      DIMENSION MSORT(5),KSORT(40)
      DIMENSION KONS(8),KODI(8),SEMBL(8),KODF(8)
      DIMENSION NOTCH(400)
      DIMENSION NFIRS(30),NLAST(30),IXPON(30),WEIGT(30)
      DIMENSION SYMBU(30),KONSO(30),NEST(30)
      COMMON NBN,NBG,NTO,NSPT,NEXPS,NPAC,NRI,NEON,NRS
      COMMON NNG,NSPTU,NBTG
      COMMON NOD,NOB,KBASI,LISTG,LISTC,LISTP
      COMMON NIN,NOUT,NODA,NODB
      COMMON NFIRS,NLAST,IXPON,WEIGT,SYMBU,KONSO,NEST,LIST
      COMMON KODI,KONS,KODF,SEMBL,MSORT,KSORT
      COMMON LIL,KIK,KOO,IZ
      COMMON NCI
      EQUIVALENCE (POLY(1,1),MIX(1))
      EQUIVALENCE (KODET(1),SMBOL(1),ITOP(1))
      EQUIVALENCE (CONS(1,1),NOTCH(1),POLYU(1,1))
      EQUIVALENCE (IXPO(1,1),SIMBD(1,1),ISET(1,1))
      EQUIVALENCE (KODE(1,1),AMAG(1,1))
      EQUIVALENCE (N(1,1),AARG(1,1))
      EQUIVALENCE (CONST(1),SIMBN(1,1))
      EQUIVALENCE (NPCOD(1),KEP(1,1))
      EQUIVALENCE (IXPOT(1),KED(1,1))
      CALL MIXL(MIX)
      MNB=0
      CALL MAINN(MIX,POLY,MNB,NPCOD,NOP,KLAS,IXPOT,CONST,KODET,IXPO,CONS
     1,KODE,N,SMBOL)
      IF(MNB)1,1,2
 2    STOP
 1    CALL SUBB(NOP,KLAS,CONST,IXPOT,KODET,NPCOD,POLY,NOTCH,ISET)
      CALL MAINE(ITOP,NB,NA,SIMBN,SIMBD,KEP,KED)
      CALL SUBE(ITOP,NB,NA,SIMBN,SIMBD,POLY,KEP,KED,JIB,JD,POLYU)
      CALL SUBD(ITOP,NB,NA,SIMBN,SIMBD,POLY,KEP,KED,JD,NSET,NK,F)
      IF(F)4,5,4
 4    CALL SUBC(ITOP,NB,NA,SIMBN,SIMBD,POLY,POLYU,KEP,KED,JIB,JD,NSET,N
     1K,AMAG,AARG)
      CALL SUBF(AMAG,AARG,NSET,NK)
 5    CALL EXIT
      END
// DUP
```

```
*STORECI                PART3 PART3
*LOCAL SUBD,(MAINE,KAND,DECOD),SUBE,(MAINN,ARRAY),(SUBB,IAND,ARRAL)
*LOCAL SUBC,MIXL,SUBF
*CCEND




// FOR MIXL
      SUBROUTINE MIXL(MIX)
C     THIS PROGRAM ORDERS SFG INFORMATION
      DIMENSION KONS(8),KODI(8),SEMBL(8),KODF(8)
      DIMENSION NFIRS(30),NLAST(30),IXPON(30),WEIGT(30)
      DIMENSION MSORT(5),KSORT(40)
      DIMENSION SYMBU(30),KONSO(30),NEST(30)
      DIMENSION MIX(30)
      COMMON NBN,NBG,NTO,NSPT,NEXPS,NPAC,NRI,NEON,NRS
      COMMON NNG,NSPTU,NBTG
      COMMON NOD,NOB,KBASI,LISTG,LISTC,LISTP
      COMMON NIN,NOUT,NODA,NODB
      COMMON NFIRS,NLAST,IXPON,WEIGT,SYMBU,KONSO,NEST,LIST
      COMMON KODI,KONS,KODF,SEMBL,MSORT,KSORT
      COMMON LIL,KIK,KOO,IZ
      COMMON NCI
      DO 87 J=1,NBG
   87 MIX(J)=J
      KONU=LIST-1
      DO 80 KON=1,KONU
      IU=KON+1
      IL=KON
      GO TO 83
   81 MXL=MIX(IL)
      MIX(IL)=MIX(IU)
      MIX(IU)=MXL
      IL=IL-1
      IU=IU-1
      IF (IL) 80,80,83
   83 MIU=MIX(IU)
      MIL=MIX(IL)
      IF(NFIRS(MIU)-NFIRS(MIL))81,89,80
   84 MXL=MIX(IL)
      MIX(IL)=MIX(IU)
      MIX(IU)=MXL
      IL=IL-1
      IU=IU-1
      IF (IL) 80,80,82
   82 MIU=MIX(IU)
      MIL=MIX(IL)
      IF(NFIRS(MIU)-NFIRS(MIL))80,89,80
   89 IF (NLAST(MIU)-NLAST(MIL))80,80,84
   80 CONTINUE
 1305 CONTINUE
      DO 602  KP1=1,NEXPS
  602  MSORT(KP1)=0
      DO 950 K02=1,NSPT
```

```
  950   KODI(KO2)=0
        DO 603 KP2=1,NTO
  603   KSORT(KP2)=0
        DO 301 INK=1,NSPT
        LIL=1
  301   KONS(INK)=0
        KIK=1
        KOO=0
        RETURN
        END
```

```
// FOR MAINN
         SUBROUTINE MAINN(MIX,POLY,MNB,NPCOD,NOP,KLAS,IXPOT,CONST,KODET,IX
       1PO,CONS,KODE,N,SMBOL)
         DIMENSION LT(15),IG(15)
         DIMENSION SMBOL(30)
        DIMENSION  N(15,15),CONS(15,15),KODE(15,15),IXPO(15,15)
         DIMENSION IFLOW(15),NP(15),KODES(15),KCNC(15)
        DIMENSION MSORT(5),KSORT(40)
        DIMENSION KONS(8),KODI(8),SEMBL(8),KODF(8)
         DIMENSION NFIRS(30),NLAST(30),IXPON(30),WEIGT(30)
         DIMENSION SYMBU(30),KONSO(30),NEST(30)
        DIMENSION POLY(5,40)
         DIMENSION NPCOD(125),IXPOT(125),CONST(125),KODET(125)
          DIMENSION MIX(30)
         COMMON NBN,NBG,NTO,NSPT,NEXPS,NPAC,NRI,NEON,NRS
         COMMON NNG,NSPTU,NBTG
         COMMON NOD,NOB,KBASI,LISTG,LISTC,LISTP
         COMMON NIN,NOUT,NODA,NODB
         COMMON NFIRS,NLAST,IXPON,WEIGT,SYMBU,KONSO,NEST,LIST
         COMMON KODI,KONS,KODF,SEMBL,MSORT,KSORT
        COMMON LIL,KIK,KOO,IZ
        COMMON NCI
         EQUIVALENCE (IG(1),IFLOW(1))
C                       PROGRAM MAIN -2
         DATA ONE/'  1'/
C                 TAKE SFG BRANCH INFORMATION AS FOUND
C                 BY SUBROUTINE AND GENERATE
C                       (1)ROUTING MATRIX INFORMATION
C                       N(J,K),AND LT(J)
C                       (2)SFG BRANCH VALUES IXPO(J,L),CONS(J,L),
C                       KODE(J,L)WHERE J=NFIRST(I),L=NLAST(I),AND
C
        IBO=0
        KO=0
        MICH=1
        K=0
        MG=1
        JLAS=1
        NCIR=1
        ININ=NIN
        INOUT=NOUT
```

```
      DO 300 INK=1,NNG
  300 IG(INK)=0
C     FIND IXPO(J,L),CONS (J,L)
      GO TO 307
  305 MG=KBASI*MG
      MICH=MICH+1
  307  IBO=IBO+1
      IF(LIST-IBO)19,4,4
 4    LOB=MIX(IBO)
      J=NFIRS(LOB)
      L=NLAST(LOB)
      IXPO(J,L)=IXPON(LOB)
      CONS(J,L)=WEIGT(LOB)
C     FIND ROUTING MATRIX
 8     IF(J-JLAS)9031,10,9031
9031  LT(JLAS)=K
      K1=K+1
      IF(JLAS-NIN)28,27,28
   27 N(JLAS,K1)=-1
      GO TO 29
   28 N(JLAS,K1)=0
   29 JLAS=JLAS+1
      K=0
      GO TO 8
   10 K=K+1
      N(J,K)=L
C     FIND KODE(J,L) AND SEMBOL(KOO)
       SMBOL(IBO)=SYMBU (LOB)
      MODE=NEST(LOB)
      IF(MODE)335,316,335
  335 IF(IG(L))5,960,5
    5 KODE(J,L)=IG(L)
      GO TO 307
  960 CONTINUE
      KPU=IBO-1
      IF(KPU)953,953,315
  315  DO 952 KP=1,KPU
       IF(SMBOL(IBO)-SMBOL(KP))952,9032,952
 9032 LOBX=MIX(KP)
      IF(KONSO(LOB)-KONSO(LOBX))952,956,952
  956 LX=NLAST(LOBX)
      KODE(J,L)=IG(LX)
      GO TO 307
  952  CONTINUE
      IF(SMBOL(IBO)-ONE)953,316,953
  953  IG(L)=MG
       KOO=KOO+1
       SEMBL(KOO)=SMBOL(IBO)
      KODE(J,L)=IG(L)
      IF(KONSO(LOB ))3,3,2
    2 KONS(KOO)=1
3     GO TO 305
  316  KODE(J,L)=0
      GO TO 307
   19 LT(JLAS)=K
      K11=K+1
```

```
      N(JLAS,K11)=0
C     C
C                        PROGRAM MAIN--3
      DO 601 KAM=1,NEXPS
      DO 601 KIM=1,NTO
  601  POLY(KAM,KIM)=0
       MPL=0
      IR=1
      NFIR=1
        KNO=0
      KODES(1)=1
      DO 2000 JS=2,NNG
 2000  KODES(JS)=2*KODES(JS-1)
      IF(LISTP)175,175,1116
 1116  WRITE(1,170)NIN,NOUT
 170   FORMAT(' PATHS FROM NODE ',I2,'TO NODE ',I2//)
       WRITE(1,1905)
 1905 FORMAT(5X,'NO.      NODE LIST')
175   IF(LISTP)1113,1113,23
 1113  K3=LT(NIN)+1
       N(NIN,K3)=0
      K2=LT(1)+1
      N(1,K2)=-1
      NIN=1
      NOUT=1
      KLAS=0
24    NFIR=0
      IF(LISTC)1209,1209,1219
1219  WRITE(1,177)
 177  FORMAT(1X,'CIRCUITS'//)
       WRITE(1,1905)
        KNO=0
 1209  CONTINUE
C                       PROGRAM MAIN--4
C                        PATH -FINDING ALGORITHM
C                        IN ADDITION,STEP PF7 CALCULATES THE COMPOSITE
C                        CODE ,CONSTANT,AND EXPONENT OF THE PATH
C     PF1(PRELIMINARY)
      DO  1112 IZO=1,NNG
 1112  IFLOW(IZO)=0
      DO 31 I1=1,NNG
  31  KONC(I1)=1
       NOP=KLAS
      KLAS=0
  23  I=2
      J=NIN
      NP(1)=NIN
      IFLOW(NIN)=1
      IFLOW(NOUT)=-1
C
  25  K=KONC(J)
C
C  PF2(FIND NEXT NODE)
      NP(I)=N(J,K)
C
C     PF3 (TEST ROUTING MATRIX)
```

```
      IF(N(J,K))100,60,34
   34 NJK=N(J,K)
      IF(IFLOW(NJK))50,38,26
   26  KONC(J)=KONC(J)+1
       GO TO 25
   38  J=NP(I)
       IFLOW(J)=1
       I=I+1
       GO TO 25
C
C      PF6(BACKSTEP)
   60 IFLOW(J)=0
      KONC(J)=I
      J=NP(I-2)
      KONC(J)=KONC(J)+1
      I=I-1
       GO TO 25
C
C    PF7(FINISH PATH)
   50 KONC(J)=KONC(J)+1
      KLAS=KLAS+1
C
C    FIND CODE FOR NODE PATH
       NPCOD(IR)=0
      ISU=I-1
      DO 2002 IS=1,ISU
      NODS=NP(IS)
 2002  NPCOD(IR)=NPCOD(IR)+KODES(NODS)
C     CALL ARRAY AND WRITE
      IF(NFIR-1)9033,179,9033
 9033 CONTINUE
      IF(LISTC)1208,1208,1206
 1206  CONTINUE
      KRU=I
  179 KNO=KNO+1
      WRITE(1,110)KNO,(NP(KR),KR=1,KRU)
  110  FORMAT(4X,I3,6X,35I3)
 1208  CONTINUE
C
       IF(NFIR-1)9034,320,9034
 9034  CONTINUE
      KODET(IR)=0
      CONST(IR)=1.
      IXPOT(IR)=0
      IEND=I
      DO 319 KEW=2,IEND
      JNODE=NP(KEW-1)
      LNODE=NP(KEW)
      KODET(IR)=KODET(IR)+KODE(JNODE,LNODE)
      CONST(IR)=CONST(IR)*CONS(JNODE,LNODE)
      IXPOT(IR)=IXPOT(IR)+IXPO(JNODE,LNODE)
  319  CONTINUE
      CONEW=CONST(IR)
      IXNEW=IXPOT(IR)
      KONEW=KODET(IR)
      CALL ARRAY(1,CONEW,IXNEW,KONEW,POLY)
```

```
  320  CONTINUE
C
C
      IR=IR+1
      IF(IR-NPAC)1361,1361,1360
 1360  WRITE(1,1362)
 1362 FORMAT(1X,' NO. OF CIRCUITS EXCEEDS L MIT-INCREASE DIMENSION'/
     1'CONTANING NPAC')
 1361   CONTINUE
      GO TO 25
C
C
C                         PROGRAM MAIN--5
C     MODIFY THE SFG BY REMOVING EVERY BRANCH CONNECTED TO THE NODE THROUGH
C     WHICH ALL CIRCUITS HAVE JUST BEEN FOUND
  100 T3=0.
      IF(NCIR-1)2010,102,2010
  102 CONTINUE
      IF(NFIR-1)104,2010,104
  103 K4=LT(NIN)+1
      N(NIN,K4)=0
      K5=LT(1)+1
      N(1,K5)=-1
      NIN=1
      NOUT=1
      GO TO 24
  104 IF(NIN-JLAS) 105,200,200
  105 NIN=J+1
      NOUT=J+1
      KONC(J)=1
      NY=LT(J)+1
      N(J,NY)=0
      DO 109 JC=NIN,JLAS
       LCOL=LT(JC)
       IF(LCOL)888,109,888
 888  IF(N(JC,LCOL)-J)109,107,109
  107 N(JC,LCOL)=0
      LT(JC)=LT(JC)-1
  109 CONTINUE
      NZ=LT(NIN)+1
      N(NIN,NZ)=-1
       NOUT=NIN
      GO TO 23
 2010 IF(NCIR-1)250,103,250
 200   GO TO 2222
250    MNB=1
 2222  RETURN
       END


// FOR SUBB
       SUBROUTINE SUBB(NOP,KLAS,CONST,IXPOT,KODET,NPCOD,POLY,NOTCH,ISET)
       DIMENSION SYMBU(30),KONSO(30),NEST(30)
```

```
      DIMENSION MSORT(5),KSORT(40)
      DIMENSION KONS(8),KODI(8),SEMBL(8),KODF(8)
       DIMENSION NFIRS(30),NLAST(30),IXPON(30),WEIGT(30)
       DIMENSION ISET(8,40)
       DIMENSION NPCOD(125),IXPOT(125),CONST(125),KODET(125)
       DIMENSION NOTCH(400)
      DIMENSION NOCTO(125),MAPO(125)
      DIMENSION NUP(125),JAC(125)
      DIMENSION POLY(5,40)
       COMMON NBN,NBG,NTO,NSPT,NEXPS,NPAC,NRI,NEON,NRS
       COMMON NNG,NSPTU,NBTG
       COMMON NOD,NOB,KBASI,LISTG,LISTC,LISTP
       COMMON NIN,NOUT,NODA,NODB
       COMMON NFIRS,NLAST,IXPON,WEIGT,SYMBU,KCNSO,NEST,LIST
       COMMON KODI,KONS,KODF,SEMBL,MSORT,KSORT
      COMMON LIL,KIK,KOO,IZ
      COMMON NCI
C     PROGRAM MAIN--6
C     FIND SECOND ORDER LOOPS
      NOL=KLAS
      KHOL=0
      DO 257 KOW=1,NPAC
 257  NOCTO(KOW)=0
      LOW1=NOP+1
      NOC=0
      NOL1=NOL-1
      DO 203 LIR1=LOW1,NOL1
      LOW2=LIR1+1
      DO 202 LIR2=LOW2,NOL
       CALL IAND(NPCOD(LIR1),NPCOD(LIR2),NAN,0)
      IF(NAN)202,201,202
  201 CONTINUE
       TCONS=CONST(LIR1)*CONST(LIR2)
      KXPO2=IXPOT(LIR1)+IXPOT(LIR2)
      KSYM2=KODET(LIR1)+KODET(LIR2)
       CALL ARRAL(2,TCONS,KXPO2,KSYM2,POLY)
      KHOL=KHOL+1
      NOC=NOC+1
      IF(NOC-NEON)1396,1396,1395
 1395 WRITE(1,1397)
 1397 FORMAT(1X,'INCREASE NEON-THE DIMENSION OF THE ARRAY NOTCH')
 1396 CONTINUE
      NOTCH(NOC)=LIR2
 202  CONTINUE
 203  NOCTO(LIR1)=NOC
      NOCTO(NOL)=NOC
C     PROGRAM MAIN  7
C     FIND LOOPS OF ORDER GREATER THAN 2
C    GENERATE THE FIRST ROW OF ISET
      NIPL=NOP+1
      KAPMA=1
      INKO=1
      DO 1170 ISC=NIPL,NOL
       INK1=NOCTO(ISC)
      IF(ISC-1)1171,1171,1172
 1172 INK2=NOCTO(ISC-1)+1
```

```
      GO TO 1173
 1171 INK2=1
 1173 IF(INK1-INK2-INKU)1170,1170,1175
 1175 INKU=INK1-INK2
 1170 CONTINUE
      IF(INKO-NCI)1391,1391,1390
 1390 WRITE(6,1392)INKO
 1392 FORMAT(1X,'INCREASE NCI  THE NO OF COPUMNS IN4D MENSION OF ISET')
 1391 CONTINUE
      DO 490 NIP=NIPL,NOL
      INKU=NOCTO(NIP)
      IF(NIP-1)210,210,211
 211   INKL=NOCTO(NIP-1)+1
      GO TO 212
  210 INKL=1
  212 CONTINUE
       IF(INKU-INKL)490,490,410
  410 JIP=0
      DO 480 INK=INKL,INKJ
      JIP=JIP+1
  480 ISET(1,JIP)=NOTCH(INK)
      MAPO(NIP)=INKU-INKL+1
C     INITIATE PROCESS FOR FINDING HIGHER ORDER LOOPS
       DO 430 KAT=1,NPAC
      JAC(KAT)=0
  430 NUP(KAT)=0
      JAC(1)=MAPO(NIP)
      KAP=2
  440 KAP=KAP-1
      IF(KAP)490,490,429
  425 KAP=KAP+1
      IF(KAP-NRI)1350,1350,1351
 1351 WRITE(1,1352)
 1352 FORMAT(1X,'INCREASE NRI- THE NO. OF ROWS IN DIMENSION OF ISET')
 1350 CONTINUE
      NUP(KAP)=0
  429 KAP1=KAP+1
      JAC(KAP1)=0
      NUP(KAP)=NUP(KAP)+1
C     LABEL LOOP OF FIRST CIRCUIT
      NAP=NUP(KAP)
      IF(KAPMA-KAP)1347,1348,1348
 1347  KAPMA=KAP
 1348 CONTINUE
      ISAT=ISET(KAP,NAP)
C     TEST LOOP OF REMAINING CKTS
      MAPU=JAC(KAP)
      MAPL=NUP(KAP)+1
      DO 435 MAPI=MAPL,MAPU
      ISOT=ISET(KAP,MAPI)
      CALL IAND(NPCOD(ISAT),NPCOD(ISOT),KAN,0)
      IF(KAN)435,455,435
  455 CONTINUE
C     WRITE
       TCONG=CONST(NIP)
       KXPOG=IXPOT(NIP)
```

```
       KSYMG=KODET(NIP)
       DO 477 LPO=1,KAP
       ITIC=NUP(LPO)
      ITUCH=ISET(LPO,ITIC)
      TCONG=TCONG*CONST(ITUCH)
      KXPOG=KXPOG+IXPOT(ITUCH)
  477 KSYMG=KSYMG+KODET(ITUCH)
       TCONG=TCONG*CONST(ISOT)
      KXPOG=KXPOG+IXPOT(ISOT)
      KSYMG=KSYMG+KODET(ISOT)
      KAPP=KAP+2
       CALL ARRAL(KAPP,TCONG,KXPOG,KSYMG,POLY)
      KHOL=KHOL+1
C     SET COUNTERS
  423 KAP1=KAP+1
      JAC(KAP1)=JAC(KAP1)+1
      JACK=JAC(KAP1)
      ISET(KAP1,JACK)=ISET(KAP,MAPI)
  435 CONTINUE
      JACK=JAC(KAP1)
      IF(JACK-2) 431,425,425
  431 IF(JAC(KAP)-NUP(KAP)-1) 440,440,429
  490 CONTINUE
       CALL ARRAL(2,1.,0,0,POLY)
250   RETURN
      END
```

```
// FOR MAINE
       SUBROUTINE MAINE(ITOP,NB,NA,SIMBN,SIMBD,KEP,KED)
       DIMENSION ITOP(40)
       DIMENSION SIMBN(40,4),SIMBD(40,4)
       DIMENSION KEP(40,4),KED(40,4)
       DIMENSION NA(40),NB(40)
      DIMENSION MSORT(5),KSORT(40)
      DIMENSION KONS(8),KODI(8),SEMBL(8),KODF(8)
       DIMENSION SYMBU(30),KONSO(30),NEST(30)
       DIMENSION NFIRS(30),NLAST(30),IXPON(30),WEIGT(30)
       COMMON NBN,NBG,NTO,NSPT,NEXPS,NPAC,NRI,NEON,NRS
       COMMON NNG,NSPTU,NBTG
       COMMON NOD,NOB,KBASI,LISTG,LISTC,LISTP
       COMMON NIN,NOUT,NODA,NODB
       COMMON NFIRS,NLAST,IXPON,WEIGT,SYMBU,KONSO,NEST,LIST
       COMMON KODI,KONS,KODF,SEMBL,MSORT,KSORT
      COMMON LIL,KIK,KOO,IZ
      COMMON NCI
      DATA SB/' 1 '/
C     PROGRAM MAIN  8
C     DECODE COMPOSITE SYMBOL CODE
C     AND ISOLATE SYMBOLS FROM
C     INVERSE SYMBOLS
      DO 693 J1=1,NTO
      DO 693 J2=1,NSPTU
```

```
      KEP(J1,J2)=1
      KED(J1,J2)=1
      SIMBN(J1,J2)=SB
  693 SIMBD(J1,J2)=SB
      DO 951 J4=1,NTO
      NA(J4)=0
  951 NB(J4)=0
C     DECODE KSORT(JZ) AND RECORD TERMS
C     CONTAINING FEEDBACK SYMBOL 'FB'
      JZU=LIL-1
      DO 646 JZ=1,JZU
      KODY=KSORT(JZ)
      ITOP(JZ)=0
      IF(KODY)715,646,715
715   CALL DECOD(KODY,JZ,ITOP)
C      ISOLATE NUM. SYMBOLS AND INVERSE SYMBOLS
C      OF KSORT(JZ)
  637 NAK=0
      NAT=0
      IF(IZ)646,646,647
  647 CONTINUE
       DO 645 NZ=1,IZ
      KOZY=KODI(NZ)
       IARG=KODF(NZ)
      IF(IARG-NRS) 1340,1340,1341
 1341 WRITE(1,1342)
 1342 FORMAT(1X,'INCREASE THE DIMENSION OF STAR')
 1340 CONTINUE
      IF(KONS(KOZY))657,657,659
  657 NAK=NAK+1
      IF(NAK-NSPTU-1)1376,1375,1375
 1375 WRITE(1,1377)
C     THE CONSTANT COEFFICIENTS IN THE TRANSFER FUNCTION ARE SEPARATED
C     INTO ARRAYS FOR THE NUMERATOR AND DENOMINATOR
 1377 FORMAT(1X,'NSPT EXCEEDS LIMIT- INCREASE DIMENSIONS, CONTAINING
     1 NSPT')
 1376 CONTINUE
      SIMBN(JZ,NAK)=SEMBL(KOZY)
       KEP(JZ,NAK)=IARG
      NA(JZ)=NA(JZ)+1
      GO TO 645
  659 NAT=NAT+1
      IF(NAT-NSPTU-1)1381,1380,1380
 1380 WRITE(1,1382)
 1382 FORMAT(1X,'NSPT EXCEEDS LIMIT-INCREASE DIMENSIONS,'
     1'CONTAINING NSPT')
 1381 CONTINUE
       SIMBD(JZ,NAT)=SEMBL(KOZY)
       KED(JZ,NAT)=IARG
      NB(JZ)=NB(JZ)+1
  645 CONTINUE
  646 CONTINUE
      RETURN
      END
```

```
// FOR SUBE
      SUBROUTINE SUBE(ITOP,NB,NA,SIMBN,SIMBD,POLY,KEP,KED,JIB,JD,POLYU)
      DIMENSION PON(4),POD(4)
      DIMENSION TEMP(5)
      DIMENSION KEP(40,4),KED(40,4)
      DIMENSION POLY(5,40),ITOP(40)
      DIMENSION NA(40),NB(40)
      DIMENSION SIMBN(40,4),SIMBD(40,4)
      DIMENSION NFIRS(30),NLAST(30),IXPON(30),WEIGT(30)
      DIMENSION KONS(8),KODI(8),SEMBL(8),KODF(8)
      DIMENSION MSORT(5),KSORT(40)
      DIMENSION SYMBU(30),KONSO(30),NEST(30)
      DIMENSION POLYU(5,40)
      COMMON NBN,NBG,NTO,NSPT,NEXPS,NPAC,NRI,NEON,NRS
      COMMON NNG,NSPTU,NBTG
      COMMON NOD,NOB,KBASI,LISTG,LISTC,LISTP
      COMMON NIN,NOUT,NODA,NODB
      COMMON NFIRS,NLAST,IXPON,WEIGT,SYMBU,KONSO,NEST,LIST
      COMMON KODI,KONS,KODF,SEMBL,MSORT,KSORT
      COMMON LIL,KIK,KOO,IZ
      COMMON NCI
C     PROGRAM MAIN  9
C     SEPARATE POLY INTO ARRAYS FOR THE NUMERATOR AND DENOMINATOR
C     OF THE TRANSFER FUNCTION
      DATA TEMP(1),TEMP(2),TEMP(3)/'    ','**2','**3'/
      DATA TEMP(4),TEMP(5)/'**4','**5'/
      DATA DASH/' /'/
  931 FORMAT(1X,50(1H*))
  930 FORMAT(//)
      DO 691 J1=1,NEXPS
      DO 691 J2=1,NTO
  691 POLYU(J1,J2)=0
      NANU=LIL-1
      KIKU=KIK-1
      DO 755 JA=1,KIKU
      JIB=0
      JD=0
      DO 755 JC=1,NANU
      IF(ITOP(JC))753,753,751
  751 JIB=JIB+1
      POLYU(JA,JIB)=POLY(JA,JC)
      GO TO 755
  753 JD=JD+1
      POLY(JA,JD)=POLY(JA,JC)
  755 CONTINUE
C     PROGRAM MAIN  10
C     MAKE POWERS OF S IN OUTPUT
C     TRANSFER FUNCTION POSITIVE
      MAXIM=0
      KARU=KIK-1
      DO 522 KAR=1,KARU
      IF(MSORT(KAR))521,522,522
  521 IF(MAXIM+MSORT(KAR))523,522,522
```

```
 523   MAXIM=-MSORT(KAR)
  522  CONTINUE
       DO 524 KIT=1,KARU
  524  MSORT(KIT)=MAXIM+MSORT(KIT)
C      MAIN PROGRAM  11
C      PRINT OUT NUMERATOR OF THE TRANSFER FUNCTION
       LUK=0
       IKU=LIL-1
       WRITE(1,931)
       WRITE(1,930)
       WRITE(1,920)
  920  FORMAT(25X,'NUMERATOR POLYNOMIAL'///)
       WRITE(1,921)
  921  FORMAT(1X,'COLUMN',12X,'SYMBOL FOR GIVEN COLUMN')
       DO 905 IK=1,IKU
       IF(ITOP(IK))905,905,901
  901  ILU=NA(IK)
       IF(ILU)710,710,711
  710  ILU=1
  711  JLU=NB(IK)
       IF(JLU)712,712,713
 712   JLU=1
 713    CONTINUE
       LUK=LUK+1
        DO 10 IL=1,ILU
        I=KEP(IK,IL)
        PON(IL)=TEMP(I)
10      CONTINUE
        DO 20 JL=1,JLU
        I=KED(IK,JL)
        POD(JL)=TEMP(I)
 20     CONTINUE
       WRITE(1,903)LUK,(SIMBN(IK,IL),PON(IL),
      1IL=1,ILU),DASH,(SIMBD(IK,JL),POD(JL),JL=1,JLU)
  903  FORMAT(1X,I5,20X,30A3)
905     CONTINUE
       WRITE(1,930)
       WRITE(1,1821)
 1821  FORMAT(1X,'POWER')
       WRITE(1,922)
  922  FORMAT(1X,'OF S',17X,'CONSTANT COEFS. IN THE POLYNOMIAL')
       LML=1
        LMU=4
       IF(JIB-LMU)820,818,818
  820  LMU=JIB
 818   WRITE(1,806)(LO,LO=LML,LMU)
  806  FORMAT(2X,7(8X,'COLUMN',I2))
       KROWU=KIK-1
       DO 808 KROW=1,KROWU
       WRITE(1,810)MSORT(KROW),(POLYU(KROW,LM),LM=LML,LMU)
  810  FORMAT(I5,'     ',7(E12.5,'     '))
  808  CONTINUE
       IF(JIB-LMU)814,814,812
  812  LML=LML+4
       LMU=LMU+4
        IF(JIB-LMU)816,818,818
```

```
  816 LMU=JIB
      GO TO 818
 814   CONTINUE
      RETURN
       END
```

```
// FOR SUBD
       SUBROUTINE SUBD(ITOP,NB,NA,SIMBN,SIMBD,POLY,KEP,KED,JD,NSET,NK,F)
C      PROGRAM MAIN  12
C      PRINT OUT DENOMINATOR OF
C      THE TRANSFER FUNCTION
       INTEGER F
       DIMENSION TEMP(5)
       DIMENSION KEP(40,4),KED(40,4)
       DIMENSION PON(4),POD(4)
       DIMENSION POLY(5,40),ITOP(40)
       DIMENSION SIMBN(40,4),SIMBD(40,4)
       DIMENSION NA(40),NB(40)
       DIMENSION NFIRS(30),NLAST(30),IXPON(30),WEIGT(30)
      DIMENSION KONS(8),KODI(8),SEMBL(8),KODF(8)
      DIMENSION MSORT(5),KSORT(40)
       DIMENSION SYMBU(30),KONSO(30),NEST(30)
       COMMON NBN,NBG,NTO,NSPT,NEXPS,NPAC,NRI,NEON,NRS
       COMMON NNG,NSPTU,NBTG
       COMMON NOD,NOB,KBASI,LISTG,LISTC,LISTP
       COMMON NIN,NOUT,NODA,NODB
       COMMON NFIRS,NLAST,IXPON,WEIGT,SYMBU,KONSO,NEST,LIST
       COMMON KODI,KONS,KODF,SEMBL,MSORT,KSORT
      COMMON LIL,KIK,KOO,IZ
      COMMON NCI
       DATA TEMP(1),TEMP(2),TEMP(3)/'    ','**2','**3'/
       DATA TEMP(4),TEMP(5)/'**4','**5'/
       DATA DASH/' /'/
  930 FORMAT(//)
  931 FORMAT(1X,50(1H*))
      LUK=0
      IKU=LIL-1
      WRITE(1,931)
      WRITE(1,930)
      WRITE(1,923)
  923 FORMAT(25X,'DENOMINATOR POLYNOMIAL'///)
      WRITE(1,924)
  924 FORMAT(1X,'COLUMN',12X,'SYMBOL FOR GIVEN COLUMN')
      DO 705 IK=1,IKU
      IF(ITOP(IK))701,701,705
  701 ILU=NA(IK)
      LUK=LUK+1
      IF(ILU)915,915,916
 915   ILU=1
  916 JLU=NB(IK)
       IF(JLU)917,917,918
  917 JLU=1
```

```
  918 CONTINUE
       DO 10 IL=1,ILU
       I=KEP(IK,IL)
       PON(IL)=TEMP(I)
10     CONTINUE
       DO 20 JL=1,JLU
       I=KED(IK,JL)
       POD(JL)=TEMP(I)
 20    CONTINUE
      WRITE(1,703)LUK,(SIMBN(IK,IL),PON(IL),
     1IL=1,ILU),DASH,(SIMBD(IK,JL),POD(JL),JL=1,JLU)
  703 FORMAT(1X,I5,20X,30A3)
  705 CONTINUE
      WRITE(1,930)
      WRITE(1,1822)
 1822 FORMAT(1X,'POWER')
      WRITE(1,925)
  925 FORMAT(1X,'  OF S  ',17X,'CONSTANT COEFS.IN THE POLYNOMIAL')
      LML=1
       LMU=4
      IF(JD-LMU)520,518,518
  520 LMU=JD
  518 WRITE(1,506) (LO,LO=LML,LMU)
  506 FORMAT(2X,7(8X,'COLUMN',I2))
      KROWU=KIK-1
      DO 508 KROW=1,KROWU
       WRITE(1,510) MSORT(KROW),(POLY(KROW,LM),LM=LML,LMU)
  510 FORMAT(I5,'   ',7(E12.5,'    '))
  508 CONTINUE
      IF(JD-LMU)514,514,512
  512 LML=LML+4
      LMU=LMU+4
      IF(JD-LMU)516,518,518
  516 LMU=JD
      GO TO 518
  514 CONTINUE
      WRITE(1,931)
       WRITE(1,930)
       READ(5,3)F
3      FORMAT(I1)
       IF(F)5,6,5
5      CONTINUE
       READ(5,2251)NSET,NK
2251  FORMAT(2I10)
      WRITE(1,4444)NSET
 4444  FORMAT(1X,'NUMBER OF SETS =',I2)
       WRITE(1,4445)NK
 4445 FORMAT(1X,'NUMBER OF FREQUENCYS =',I2)
       WRITE(1,930)
6      CONTINUE
       RETURN
       END
```

```
// FOR SUBC
      SUBROUTINE SUBC(ITOP,NB,NA,SIMBN,SIMBD,POLY,POLYU,KEP,KED,JIB,JD,
     1NSET,NK,AMAG,AARG)
C     THIS SUBROUTINE FINDS THE AMPLITUDE AND PHASE
C     OF NETWORK FUNCTION
      DIMENSION POLYU(5,40)
      DIMENSION NA(40),NB(40)
      DIMENSION SIMBN(40,4),SIMBD(40,4)
      DIMENSION POLY(5,40),ITOP(40)
      DIMENSION KEP(40,4),KED(40,4)
      DIMENSION AMAG(10,10),AARG(10,10)
      DIMENSION VALX(8)
      DIMENSION DN(40)
      DIMENSION PN(40)
      DIMENSION NFIRS(30),NLAST(30),IXPON(30),WEIGT(30)
      DIMENSION KONS(8),KODI(8),SEMBL(8),KODF(8)
      DIMENSION MSORT(5),KSORT(40)
      DIMENSION SYMBU(30),KONSO(30),NEST(30)
      COMMON NBN,NBG,NTC,NSPT,NEXPS,NPAC,NRI,NEON,NRS
      COMMON NNG,NSPTU,NBTG
      COMMON NOD,NOB,KBASI,LISTG,LISTC,LISTP
      COMMON NIN,NOUT,NODA,NODB
      COMMON NFIRS,NLAST,IXPON,WEIGT,SYMBU,KONSO,NEST,LIST
      COMMON KODI,KONS,KODF,SEMBL,MSORT,KSORT
      COMMON LIL,KIK,KOO,IZ
      COMMON NCI
      EQUIVALENCE (PN(1),DN(1))
      BMAG(W)=SQRT((TRPN**2+W**2*TGPN**2)/(TRDN**2+W**2*TGDN**2))
      ARG(W)=ATAN(W*(TRDN*TGPN-TGDN*TRPN)/(TRPN*TRDN+W**2*TGPN*TGDN))
      SI=4.7136
      IKU=LIL-1
      PI2=6.28318
      WRITE(1,931)
      DO 2254 K=1,NSET
      WRITE(1,930)
930   FORMAT(/)
931   FORMAT(1X,50(1H*))
      WRITE(1,5561)K
5561  FORMAT(1X,'SET NUMBER =',I3)
      WRITE(1,930)
      IF(K-2)50,51,50
51    PAUSE
50    CONTINUE
      L=1
      LL=1
      M=0
7     M=M+1
      SEMBL(M)=SIMBN(L,LL)
      IF(K-1)52,53,52
53    CONTINUE
      WRITE(1,1)M,SEMBL(M)
1     FORMAT(10X,'SYMBOL (',I2,') =',A3)
      PAUSE
52    CONTINUE
      READ(5,2)VALX(M)
```

```
  2     FORMAT(E12.5)
        WRITE(1,54)M,VALX(M)
 54     FORMAT(10X,'SYMBOL (',I2,') =',E12.5)
        SIMBN(L,LL)=VALX(M)
 8      IF(LL-NSPTU)3,4,4
 3      LL=LL+1
 11     DO 5KM=1,M
        IF(SIMBN(L,LL)-SEMBL(KM))5,6,5
 5      CONTINUE
        GO TO 7
 6      SIMBN(L,LL)=VALX(KM)
        GO TO 8
 4      IF(L-IKU)9,10,10
 9      L=L+1
        LL=1
        GO TO 11
 10     CONTINUE
        L=1
        LL=1
        GO TO 21
 17     M=M+1
        SEMBL(M)=SIMBD(L,LL)
        IF(K-1)55,56,55
 56     CONTINUE
        WRITE(1,1)M,SEMBL(M)
        PAUSE
 55     CONTINUE
        READ(5,2)VALX(M)
        WRITE(1,54)M,VALX(M)
        SIMBD(L,LL)=VALX(M)
 18     IF(LL-NSPTU)13,14,14
 13     LL=LL+1
 21     DO 15 KM=1,M
        IF(SIMBD(L,LL)-SEMBL(KM))15,16,15
 15     CONTINUE
        GO TO 17
 16     SIMBD(L,LL) =VALX(KM)
        GO TO 18
 14     IF(L-IKU)19,20,20
 19     L=L+1
        LL=1
        GO TO 21
 20     CONTINUE
        TRPN=0.
        TGPN=0.
        TRDN=0.
        TGDN=0.
        DO 2115 I=1,NTO
        PN(I)=0.
 2115   CONTINUE
        LUK=0
        DO 905 IK=1,IKU
        IF(ITOP(IK))905,905,901
 901    ILU=NA(IK)
        IF(ILU)710,710,711
 710    ILU=1
```

```
  711 JLU=NB(IK)
      IF(JLU)712,712,713
 712  JLU=1
 713   CONTINUE
      LUK=LUK+1
       PNP=1.
       PND=1.
       DO 2117 IL=1,ILU
       PNP=(SIMBN(IK,IL))**KEP(IK,IL)*PNP
 2117  CONTINUE
      DO 2227 JL=1,JLU
       PND=(SIMBD(IK,JL)**KED(IK,JL))*PND
2227  CONTINUE
      PN(LUK)=PNP/PND
905    CONTINUE
      KROWU=KIK-1
      DO 808 KROW=1,KROWU
       DO 2225 LM=1,JIB
       IF(MSORT(KROW))2118,2119,2118
 2119  TRPN=TRPN+POLYU(KROW,LM)*PN(LM)
       GO TO 2225
 2118  KROWW=MSORT(KROW)/2
       IF(KROWW*2-MSORT(KROW))2220,2221,2220
2221    TRPN=TRPN+POLYU(KROW,LM)*PN(LM)*(-1)**KROWW
       GO TO 2225
 2220 TGPN=TGPN+POLYU(KROW,LM)*PN(LM)**(-1)**KROWW
 2225 CONTINUE
  808 CONTINUE
       DO 2357 I=1,NTO
       DN(I)=0.
 2357  CONTINUE
       LUK=0
       IKU=LIL-1
      DO 705 IK=1,IKU
      IF(ITOP(IK))701,701,705
  701 ILU=NA(IK)
      IF(ILU)915,915,916
 915   ILU=1
  916 JLU=NB(IK)
       IF(JLU)917,917,918
  917 JLU=1
  918 CONTINUE
      LUK=LUK+1
       PNP=1.
       PND=1.
       DO 2167 IL=1,ILU
       PNP=(SIMBN(IK,IL))**KEP(IK,IL)*PNP
 2167  CONTINUE
       DO 2238 JL=1,JLU
       PND=(SIMBD(IK,JL)**KED(IK,JL))*PND
 2238  CONTINUE
       DN(LUK)=PNP/PND
  705 CONTINUE
      KROWU=KIK-1
      DO 508 KROW=1,KROWU
      DO 2267 LM=1,JD
```

```
      IF(MSORT(KROW))2228,2229,2228
 2229  TRDN=TRDN+POLY(KROW,LM)*DN(LM)
      GO TO 2267
2228   KROWW=MSORT(KROW)/2
       IF(KROWW*2-MSORT(KROW))2350,2271,2350
 2271  TRDN=TRDN+POLY(KROW,LM)*DN(LM)*(-1)**KROWW
      GO TO 2267
2350   TGDN=TGDN+POLY(KROW,LM)*DN(LM)*(-1)**KROWW
2267  CONTINUE
  508 CONTINUE
       WRITE(1,930)
       WRITE(1,5555)TRPN
 5555  FORMAT(1X,'REAL VALUE OF NUMERATOR =',E12.5)
       WRITE(1,5556)TGPN
 5556  FORMAT(1X,'IMAGINARY VALUE OF NUMERATOR =',E12.5)
       WRITE(1,5557)TRDN
 5557  FORMAT(1X,'REAL VALUE OF DENOMINATOR =',E12.5)
       WRITE(1,5558)TGDN
 5558  FORMAT(1X,'IMAGINARY VALUE OF DENOMINATOR =',E12.5)
       WRITE(1,930)
       WRITE(1,5559)
5559   FORMAT(10X,'FREQUENCY          AMPLITUDE          PHASE ANGLE')
       L=NK+1
       DO 2356 KK=1,L
       KK1=KK-1
       IF(KK-1)2351,2352,2351
2352   W=PI2
       F=1.
       GO TO 2353
 2351  W=PI2*10.**KK1
       F=10.**KK1
2353   CONTINUE
       AMAG(K,KK)=BMAG(W)
       AARG(K,KK)=ARG(W)
       WRITE(1,5560)F,AMAG(K,KK),AARG(K,KK)
 5560  FORMAT(9X,E12.5,8X,E12.5,8X,E12.5)
 2356  CONTINUE
       WRITE(1,930)
       WRITE(1,931)
 2254  CONTINUE
       RETURN
       END


// FOR SUBF
       SUBROUTINE SUBF(AMAG,AARG,NSET,NK)
C      THIS SUBROUTINE PLOTS THE FREQUENCY RESPONSE
       DIMENSION AMAG(10,10),AARG(10,10)
       DIMENSION XX(10,10)
       SI=4.7136
C     BRING PLOTTER PEN TO EXETREME RIGHT-POSITION
       L=NK+1
       MM=1
```

```
40     CALL SCALF(1.0,1.0,0.,-9.35)
       CALL FPLOT(1,0.,0.)
       CALL FGRID(3,0.,0.,1.,NK)
       CALL FGRID(0,0.,0.,.25,24)
       NN=NK+1
       DO 2260 I=1,NN
       AX=-.3
       AY=I-1
       CALL FCHAR(AX,-AY,0.1,0.1,SI)
       WRITE(7,4)
 4     FORMAT('10')
       AY1=I-1+.2
       AX1=-.2
       K=I-1
       CALL FCHAR(AX1,-AY1,0.07,0.07,SI)
       WRITE(7,6)K
 6     FORMAT(I2)
2260   CONTINUE
       DO 2270 I=1,NK
       AI=I-1
       DO 2270 J=2,9
       AJ=J
       U=AI+ALOG(AJ)/2.303
       CALL FGRID(3,0.,-U,0.,0)
2270   CONTINUE
       ND=NK/2
       AY=ND
       AY1=AY-1.
       CALL FCHAR(-.6,-AY1,.25,.175,SI)
       WRITE(7,7)
 7     FORMAT(1X,'FREQUENCY')
       CALL FCHAR(2.5,0.8,.25,.175,0.)
       IF(MM-1)34,35,34
35     WRITE(7,8)
 8     FORMAT(1X,'AMPLITUDE')
       DO 37 K=1,NSET
       DO 37 KK=1,L
       XX(K,KK)=AMAG(K,KK)
37     CONTINUE
       GO TO 36
34     WRITE(7,9)
 9     FORMAT(1X,'PHASE ANGLE')
       DO 38 K=1,NSET
       DO 38 KK=1,L
       XX(K,KK)=AARG(K,KK)
38     CONTINUE
36     CONTINUE
       X2=2.3
       DO 12 KJ=1,NSET
       CALL FPLOT(-2,X2,1.35)
       CALL POINT(KJ)
       CALL FCHAR(X2,1.0,0.075,0.075,SI)
       CALL FCHAR(X2,1.25,0.1,0.1,SI)
       WRITE(7,11)KJ
11     FORMAT(1X,'SET',I2)
       X2=X2-.15
```

```
12      CONTINUE
        TMAX=XX(1,1)
        TMIN=XX(1,1)
        DO 2258 K=1,NSET
        DO 2258 KK=1,L
        IF(XX(K,KK)-TMAX)2263,2263,2261
 2263   IF(TMIN-XX(K,KK))2258,2258,2262
 2261   TMAX=XX(K,KK)
        GO TO 2258
2262    TMIN=XX(K,KK)
2258    CONTINUE
        IF(MM-1)2578,2579,2578
2579    WRITE(1,2580)TMAX
 2580   FORMAT(1X,'MAX. VALUE OF AMPLITUDE =',E12.5)
        WRITE(1,2583)TMIN
2583    FORMAT(1X,'MIN. VALUE OF AMPLITUDE =',E12.5)
        GO TO 2582
2578    WRITE(1,2581)TMAX
 2581 FORMAT(1X,'MAX. VALUE OF PHASE-ANGLE =',E12.5)
        WRITE(1,2584)TMIN
2584    FORMAT(1X,'MIN. VALUE OF PHASE-ANGLE =',E12.5)
2582    CONTINUE
        IK=0
        SC=(TMAX-TMIN)/6.
        DO 2280 I=1,7
       AX=I-1
        CALL FCHAR(AX,.8,.1,.1,SI)
        AA=TMIN+SC*AX
        WRITE(7,28)AA
28      FORMAT(F7.2)
2280    CONTINUE
        CALL FPLOT(-1,0.,0.)
        AC=1./SC
        CALL SCALF(AC,1.,TMIN,0.)
        DO 2290 I=1,NSET
        IK=IK+1
        NN=NK+1
       DO 2300 J=1,NN
        X=XX(I,J)
        Y=FLOAT(J-1)
        CALL FPLOT(-2,X,-Y)
        CALL POINT(IK)
2300   CONTINUE
        CALL FPLOT(1,0.,0.)
 2290   CONTINUE
        IF(MM-1)31,30,31
30      X=TMAX+SC*2.
        CALL FPLOT(1,X,-9.35)
        MM=MM-1
        GO TO 40
31      WRITE(1,931)
        WRITE(1,930)
930     FORMAT(//)
931     FORMAT(1X,50(1H*))
        RETURN
        END
```

```
// FOR DECOD
      SUBROUTINE DECOD(KODY,JZ,ITOP)
C*********************************************************************
C  THE FOLLOWING ARRAYS ARE ASSOCIATED WITH THE NETWORK
C  CHARACTERISTICS NSPT, AND NTO (DEFINED IN PROGRAM MAIN -1)
      DIMENSION ITOP(40)
      DIMENSION KONS(8),KODI(8),SEMBL(8),KODF(8)
      DIMENSION MSORT(5),KSORT(40)
       DIMENSION SYMBU(30),KONSO(30),NEST(30)
       DIMENSION NFIRS(30),NLAST(30),IXPON(30),WEIGT(30)
C*********************************************************************
       COMMON NBN,NBG,NTO,NSPT,NEXPS,NPAC,NRI,NEON,NRS
       COMMON NNG,NSPTU,NBTG
       COMMON NOD,NOB,KBASI,LISTG,LISTC,LISTP
       COMMON NIN,NOUT,NODA,NODB
       COMMON NFIRS,NLAST,IXPON,WEIGT,SYMBU,KONSO,NEST,LIST
       COMMON KODI,KONS,KODF,SEMBL,MSORT,KSORT
      COMMON LIL,KIK,KOO,IZ
      COMMON NCI
      DATA FB/' FB'/
      IZ=0
      M=KBASI-1
      DO 3 J=1,KOO
      CALL KAND(KODY,M,IPOWE,1)
      IF(IPOWE)3,3,2
 2    IF(SEMBL(J)+FB)9082,4,9082
 9082 CONTINUE
      IZ=IZ+1
      IF (IZ-NSPT-1) 1371,1370,1370
 1370 WRITE (1,1372)
 1372 FORMAT (1X,'  NO. OF SYMBOLS PER TERM EXCEEDS OUTPUT-INCREASE ,
     1   DIMENSIONS CONTAINING NSPT')
 1371 CONTINUE
       KODF(IZ)=IPOWE
      KODI(IZ)=J
      GO TO 3
    4 ITOP(JZ)=1
 3     KODY=KODY/BBASI
      RETURN
      END


// FOR ARRAY
       SUBROUTINE ARRAY(JSIG,XCON,JXPO,JKOD,POLY)
       DIMENSION NFIRS(30),NLAST(30),IXPON(30),WEIGT(30)
       DIMENSION SYMBU(30),KONSO(30),NEST(30)
      DIMENSION POLY(5,40)
      DIMENSION MSORT(5),KSORT(40)
```

```
      DIMENSION KONS(8),KODI(8),SEMBL(8),KODF(8)
       COMMON NBN,NBG,NTO,NSPT,NEXPS,NPAC,NRI,NEON,NRS
       COMMON NNG,NSPTU,NBTG
       COMMON NOD,NOB,KBASI,LISTG,LISTC,LISTP
       COMMON NIN,NOUT,NODA,NODB
       COMMON NFIRS,NLAST,IXPON,WEIGT,SYMBU,KCNSO,NEST,LIST
       COMMON KODI,KONS,KODF,SEMBL,MSORT,KSORT
      COMMON LIL,KIK,KOD,IZ
      COMMON NCI
C*********************************************************************
C  THE FOLLOWING ARRAYS ARE ASSOCIATED WITH THE NETWORK
C  CHARACTERISTICS NTO, AND NEXPS (DEFINED IN PROGRAM MAIN-1)
      MMX=0
      NNX=0
      IF (KIK-1) 3,22,3
 3    MMU=KIK-1
      DO 2 MM=1,MMU
      MMX=MMX+1
      IF (JXPO-MSORT(MM)) 2,10,2
    2 CONTINUE
   22 MSORT(KIK)=JXPO
      MMX=KIK
      KIK=KIK+1
      IF (KIK-NEXPS-1) 1386,1385,1385
 1385 WRITE(1,1387)
 1387 FORMAT (1X,'   S-POWER EXCEEDS L+M+T-+NC-EASE D+MENS+ONS ,
     /    CONTAINING NEXPS')
 1386 CONTINUE
   10 IF (LIL-1) 11,24,11
   11 NNU=LIL-1
      DO 12 NN=1,NNU
      NNX=NNX+1
      IF (JKOD-KSORT(NN)) 12,20,12
   12 CONTINUE
 24   KSORT(LIL)=JKOD
      NNX=LIL
      LIL=LIL+1
      IF (LIL-NTO-1) 1367,1365,1365
 1365 WRITE (1,1366)
 1367 CONTINUE
 1366 FORMAT (1X,'  NO. OF TERMS IN OUTPUT EXCEEDS LIMIT-INCREASE
     1    DIMENSIONS CONTAINING NTO')
   20 POLY(MMX,NNX)=POLY(MMX,NNX)+XCON*(-1.)**JSIG
       RETURN
      END


// FOR ARRAL
       SUBROUTINE ARRAL(JSIG,XCON,JXPO,JKOD,POLY)
       DIMENSION NFIRS(30),NLAST(30),IXPON(30),WEIGT(30)
       DIMENSION SYMBU(30),KONSO(30),NEST(30)
      DIMENSION POLY(5,40)
      DIMENSION MSORT(5),KSORT(40)
```

```
      DIMENSION KONS(8),KODI(8),SEMBL(8),KODF(8)
       COMMON NBN,NBG,NTO,NSPT,NEXPS,NPAC,NRI,NEON,NRS
       COMMON NNG,NSPTU,NBTG
       COMMON NOD,NOB,KBASI,LISTG,LISTC,LISTP
       COMMON NIN,NOUT,NODA,NODB
       COMMON NFIRS,NLAST,IXPON,WEIGT,SYMBU,KCNSO,NEST,LIST
       COMMON KODI,KONS,KODF,SEMBL,MSORT,KSORT
      COMMON LIL,KIK,KOO,IZ
      COMMON NCI
C*********************************************************************
C  THE FOLLOWING ARRAYS ARE ASSOCIATED WITH THE NETWORK
C  CHARACTERISTICS NTO, AND NEXPS (DEFINED IN PROGRAM MAIN-1)
      MMX=0
      NNX=0
      IF (KIK-1) 3,22,3
 3    MMU=KIK-1
      DO 2 MM=1,MMU
      MMX=MMX+1
      IF (JXPO-MSORT(MM)) 2,10,2
    2 CONTINUE
   22 MSORT(KIK)=JXPO
      MMX=KIK
      KIK=KIK+1
      IF (KIK-NEXPS-1) 1386,1385,1385
 1385 WRITE(1,1387)
 1387 FORMAT (1X,'    S-POWER EXCEEDS L+M+T-+NC-EASE D+MENS+ONS ,
     /    CONTAINING NEXPS')
 1386 CONTINUE
   10 IF (LIL-1) 11,24,11
   11 NNU=LIL-1
      DO 12 NN=1,NNU
      NNX=NNX+1
      IF (JKOD-KSORT(NN)) 12,20,12
   12 CONTINUE
 24   KSORT(LIL)=JKOD
      NNX=LIL
      LIL=LIL+1
      IF (LIL-NTO-1) 1367,1365,1365
 1365 WRITE (1,1366)
 1367 CONTINUE
 1366 FORMAT (1X,'  NO. OF TERMS IN OUTPUT EXCEEDS LIMIT-INCREASE ,
     1    DIMENSIONS CONTAINING NTO')
   20 POLY(MMX,NNX)=POLY(MMX,NNX)+XCON*(-1.)**JSIG
       RETURN
      END


// FOR IAND
      SUBROUTINE IAND(MX,NX,MN,IFLAG)
C       THIS SUBROUTINE FINDS THE 'AND' OPERATION OF
C       TWO DECIMAL NUMBERS
       DIMENSION SYMBU(30),KONSO(30),NEST(30)
      DIMENSION KONS(8),KODI(8),SEMBL(8),KODF(8)
      DIMENSION MSORT(5),KSORT(40)
       DIMENSION NFIRS(30),NLAST(30),IXPON(30),WEIGT(30)
```

```
      COMMON NBN,NBG,NTO,NSPT,NEXPS,NPAC,NRI,NEON,NRS
      COMMON NNG,NSPTU,NBTG
      COMMON NOD,NOB,KBASI,LISTG,LISTC,LISTP
      COMMON NIN,NOUT,NODA,NODB
      COMMON NFIRS,NLAST,IXPON,WEIGT,SYMBU,KONSO,NEST,LIST
      COMMON  KODI,KONS,KODF,SEMBL,MSORT,KSORT
      COMMON LIL,KIK,KOD,IZ
      COMMON NCI
      M=MX
      N=NX
      IF(IFLAG)9083,5,9083
 9083 CONTINUE
      KBA=KBASI
      DO 6 K=1,64
      KBA=KBA/2
      IF(KBA-1)6,8,6
 6    CONTINUE
 8    LAST=K
      GO TO 7
    5 LAST=25
C     25 IS THE MAXIMUM NO. OF NODES IN SFG. CHANGE AS NEEDED
    7 MN=0
      NTHTW=1
      DO 10 I=1,LAST
      NTHTW=NTHTW*2
      NTEMP=N/2
      NTEMP=NTEMP*2
      IF(N-NTEMP)3,1,3
 3    MTEMP=M/2
      MTEMP=MTEMP*2
      IF(M-MTEMP)2,1,2
    2 MN=MN+NTHTW/2
      IF(IFLAG)1,4,1
1     M=M/2
      N=N/2
 10   CONTINUE
 4    RETURN
      END



// FOR KAND
      SUBROUTINE KAND(MX,NX,MN,IFLAG)
C     THIS SUBROUTINE FINDS THE 'AND' OPERATION OF
C     TWO DECIMAL NUMBERS
      DIMENSION SYMBU(30),KONSO(30),NEST(30)
      DIMENSION KONS(8),KODI(8),SEMBL(8),KODF(8)
      DIMENSION MSORT(5),KSORT(40)
      DIMENSION NFIRS(30),NLAST(30),IXPON(30),WEIGT(30)
      COMMON NBN,NBG,NTO,NSPT,NEXPS,NPAC,NRI,NEON,NRS
      COMMON NNG,NSPTU,NBTG
      COMMON NOD,NOB,KBASI,LISTG,LISTC,LISTP
      COMMON NIN,NOUT,NODA,NODB
      COMMON NFIRS,NLAST,IXPON,WEIGT,SYMBU,KONSO,NEST,LIST
```

```
      COMMON  KUDI,KONS,KDDF,SEMBL,MSORT,KSORT
      COMMON LIL,KIK,KOD,IZ
      COMMON NCI
      M=MX
       N=NX
      IF(IFLAG)9083,5,9083
 9083 CONTINUE
      KBA=KBASI
      DO 6 K=1,64
      KBA=KBA/2
       IF(KBA-1)6,8,6
 6    CONTINUE
 8    LAST=K
      GO TO 7
    5 LAST=25
C     25 IS THE MAXIMUM NO. OF NODES IN SFG. CHANGE AS NEEDED
    7 MN=0
       NTHTW=1
      DO 10 I=1,LAST
       NTHTW=NTHTW*2
      NTEMP=N/2
       NTEMP=NTEMP*2
      IF(N-NTEMP)3,1,3
 3     MTEMP=M/2
      MTEMP=MTEMP*2
       IF(M-MTEMP)2,1,2
    2 MN=MN+NTHTW/2
      IF(IFLAG)1,4,1
1      M=M/2
        N=N/2
 10    CONTINUE
 4    RETURN
       END
```